Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Alnair Finance Protocol
Platform: Astar Network
Language: Solidity
Date: March 18th, 2022

Table of contents

Introduction 4
Project Background 4
Audit Scope 5
Claimed Smart Contract Features 6
Audit Summary 8
Technical Quick Stats 9
Code Quality 10
Documentation 10
Use of Dependencies 10
AS-IS overview 11
Severity Definitions 19
Audit Findings 20
Conclusion 26
Our Methodology 27
Disclaimers 29
Appendix

- Code Flow Diagram 30
- Slither Results Log 37
- Solidity static analysis 46
- Solhint Linter 57

Email: audit@EtherAuthority.io

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

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the catcoin.club team to perform the Security audit of the StakingToken Protocol smart contracts code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on March 18th, 2022.

The purpose of this audit was to address the following:

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

Project Background

Alnair Finance Contract is an ERC20 smart contract, having functions like add and set pool, withdraw, deposit, mint, burn, add and remove minter, farm, rescueETH, rescueToken, claim, addLiquidity, swapTokensForNative, openTrading, etc.

Email: audit@EtherAuthority.io

Audit scope

Name	Code Review and Security Analysis Report for Alnair Finance Protocol Smart Contracts
Platform	**Astar / Solidity**
File 1	NativeFarm.sol
File 1 MD5 Hash	0E9C486B58E8A44B06E07241FF343134
Updated File 1 MD5 Hash	72B94F74B3DCD27B245F8D3CE6BE2E1D
File 2	NATIVEToken.sol
File 2 MD5 Hash	D65B83BE0D4C600DAAB22C510966AAAD
Updated File 2 MD5 Hash	76FDBE2992836A98F49446806DB23FBD
File 3	Strategy.sol
File 3 MD5 Hash	C82FB44BB4B947B7CBD1A9F0E6D662E9
Updated File 3 MD5 Hash	C7AE3DE77DB290B3241B400912BEDD69
File 4	Strategy_Arthswap.sol
File 4 MD5 Hash	AF6375A925CCD7704089DD2A3BFB7BF1
Updated File 4 MD5 Hash	C628D4C33FD0F0E8A33B7BA4F1D5AED6
File 5	TimelockController.sol
File 5 MD5 Hash	437D66C19D3CDB463EEF42159BDF703A
Updated File 5 MD5 Hash	AE79CAAFE720919B4F8E7DEDFA4E221E
File 6	xALNR.sol
File 6 MD5 Hash	EA187A7209A1E6D5434B34F75F1B5A77
Updated File 6 MD5 Hash	A6252B5418D0261CEBF1DABED8098061
File 7	Nika.sol
File 7 MD5 Hash	320485B3559976FF9B4990B40250ECCD
Audit Date	March 18th,2022

Email: audit@EtherAuthority.io

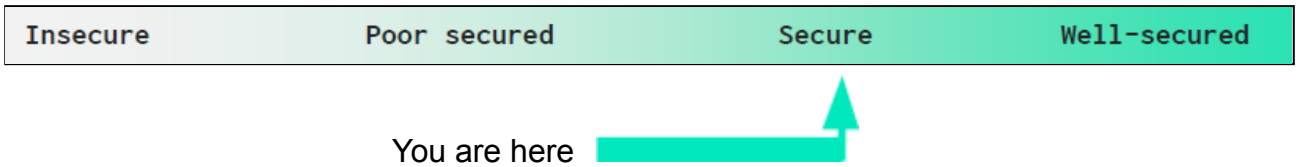
Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 NativeFarm.sol • Natives per second: 0.4 • Withdrawal FeeFactor: 10 Thousand • Withdrawal Fee Factor Maximum: 10 Thousand • Withdrawal Fee Factor Lower Limit: 9500 • Dev NATIVE Reward: 1500	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 2 NATIVEToken.sol • Decimals: 18 • Initial Supply: 2.8 Million	**YES, This is valid.**
File 3 Strategy.sol • Router Deadline Duration: 3% • Controller Fee: 2% • Controller Fee Maximum: 100% • Controller Fee Upper Limit: 3% • Buy Back Rate: 2% • Buy Back Rate Maximum: 100% • Buy Back Rate Upper Limit: 8% • Entrance Fee Factor Maximum: 100% • Entrance Fee Factor Lower Limit: 0.5% • Deposit Fee Factor: 4% • Deposit Fee Factor Maximum: 100% • Deposit Fee Factor Lower Limit: 9500 • Withdraw Fee Factor Maximum: 100% • Withdraw Fee Factor Lower Limit: 0.5% • Slippage Factor: 5% • Slippage Factor Upper Limit: 995	**YES, This is valid.** **Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 4 Nika.sol • Name: Nika	**YES, This is valid.**

• Symbol: NIKA • Decimals: 18 • Treasury Fee BPS: 2% • Liquidity Fee BPS: 2% • Dividend Fee BPS: 4% • Total Fee BPS: 8% • Swap Tokens At Amount: 1% • Maximum Tx BPS: 49 • Maximum Wallet BPS: 2%	**Owner authorized wallet can set some percentage value and we suggest handling the private key of that wallet securely.**
File 5 Strategy_Arthswap.sol • The Strategy_Arthswap contract can access strategy contracts for all functions.	**YES, This is valid.**
File 6 TimelockController.sol • Minimum Delay: 60 second • Minimum Delay Reduced: 30 second	**YES, This is valid.**
File 7 xALNR.sol • Name: Alnr Staking Token • Symbol: xALNR • Decimals: 18	**YES, This is valid.**

Audit Summary

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



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

We found 0 critical, 0 high, 2 medium and 6 low and some very low level issues.

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

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	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	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Moderated
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

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

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

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

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

Documentation

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

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

Use of Dependencies

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

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

AS-IS overview

NativeFarm.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	poolLength	external	Passed	No Issue
3	add	external	Critical operation lacks event log	Refer Audit Findings
4	set	external	Critical operation lacks event log	Refer Audit Findings
5	getMultiplier	read	Passed	No Issue
6	pendingNATIVE	external	Passed	No Issue
7	stakedWantTokens	external	Passed	No Issue
8	massUpdatePools	write	Passed	No Issue
9	updatePool	write	Critical operation lacks event log	Refer Audit Findings
10	deposit	external	Passed	No Issue
11	withdraw	write	Passed	No Issue
12	withdrawAll	external	Passed	No Issue
13	emergencyWithdraw	external	Passed	No Issue
14	safeNATIVETransfer	internal	Passed	No Issue
15	setNATIVEPerSecond	external	access only Owner	No Issue
16	inCaseTokensGetStuck	external	access only Owner	No Issue
17	setwithdrawalFeeFactor	external	access only Owner	No Issue
18	setDevaddr	write	Passed	No Issue
19	owner	read	Passed	No Issue
20	onlyOwner	modifier	Passed	No Issue
21	renounceOwnership	write	access only Owner	No Issue
22	transferOwnership	write	access only Owner	No Issue
23	nonReentrant	modifier	Passed	No Issue

NATIVEToken.sol

Functions

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

10	totalSupply	read	Passed	No Issue
11	balanceOf	read	Passed	No Issue
12	transfer	write	Passed	No Issue
13	allowance	read	Passed	No Issue
14	approve	write	Passed	No Issue
15	transferFrom	write	Passed	No Issue
16	increaseAllowance	write	Passed	No Issue
17	decreaseAllowance	write	Passed	No Issue
18	_transfer	internal	Passed	No Issue
19	_mint	internal	Passed	No Issue
20	_burn	internal	Passed	No Issue
21	_approve	internal	Passed	No Issue
22	_setupDecimals	internal	Passed	No Issue
23	_beforeTokenTransfer	internal	Passed	No Issue
24	mint	external	access only onlyMinter	No Issue
25	addMinter	external	Function input parameters lack of check	Refer Audit Findings
26	removeMinter	external	Function input parameters lack of check	Refer Audit Findings

Strategy.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	nonReentrant	modifier	Passed	No Issue
7	paused	read	Passed	No Issue
8	whenNotPaused	modifier	Passed	No Issue
9	whenPaused	modifier	Passed	No Issue
10	_pause	internal	Passed	No Issue
11	_unpause	internal	Passed	No Issue
12	onlyAllowGov	modifier	Passed	No Issue
13	deposit	write	Unused function parameter	Refer Audit Findings
14	farm	write	Passed	No Issue
15	_farm	internal	Passed	No Issue
16	_unfarm	internal	Passed	No Issue

17	withdraw	write	Unused function parameter	Refer Audit Findings
18	_harvest	internal	Passed	No Issue
19	earn	write	Passed	No Issue
20	buyBack	internal	Passed	No Issue
21	distributeFees	internal	Passed	No Issue
22	convertDustToEarned	write	Passed	No Issue
23	pause	write	access only Allow Gov	No Issue
24	unpause	write	access only Allow Gov	No Issue
25	setEntranceFeeFactor	write	access only Allow Gov	No Issue
26	setControllerFee	write	access only Allow Gov	No Issue
27	setGov	write	access only Allow Gov	No Issue
28	setDepositFeeFactor	write	access only Allow Gov	No Issue
29	setWithdrawFeeFactor	write	access only Allow Gov	No Issue
30	setbuyBackRate	write	access only Allow Gov	No Issue
31	setBuybackRouterAddress	write	access only Allow Gov	No Issue
32	inCaseTokensGetStuck	write	access only Allow Gov	No Issue
33	_wrapFTM	internal	access only Allow Gov	No Issue
34	_safeSwap	internal	Passed	No Issue

Strategy_Arthswap.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	owner	read	Passed	No Issue
3	onlyOwner	modifier	Passed	No Issue
4	renounceOwnership	write	access only Owner	No Issue
5	transferOwnership	write	access only Owner	No Issue
6	nonReentrant	modifier	Passed	No Issue
7	paused	read	Passed	No Issue
8	whenNotPaused	modifier	Passed	No Issue
9	whenPaused	modifier	Passed	No Issue
10	_pause	internal	Passed	No Issue
11	_unpause	internal	Passed	No Issue
12	onlyAllowGov	modifier	Passed	No Issue
13	deposit	write	access only Owner	No Issue

14	farm	write	Passed	No Issue
15	_farm	internal	Passed	No Issue
16	_unfarm	internal	Passed	No Issue
17	withdraw	write	access only Owner	No Issue
18	_harvest	internal	Passed	No Issue
19	earn	write	Passed	No Issue
20	buyBack	internal	Passed	No Issue
21	distributeFees	internal	Passed	No Issue
22	convertDustToEarned	write	Passed	No Issue
23	pause	write	access only Allow Gov	No Issue
24	unpause	write	access only Allow Gov	No Issue
25	setEntranceFeeFactor	write	access only Allow Gov	No Issue
26	setControllerFee	write	access only Allow Gov	No Issue
27	setGov	write	access only Allow Gov	No Issue
28	setDepositFeeFactor	write	access only Allow Gov	No Issue
29	setWithdrawFeeFactor	write	access only Allow Gov	No Issue
30	setbuyBackRate	write	access only Allow Gov	No Issue
31	setBuybackRouterAddress	write	access only Allow Gov	No Issue
32	inCaseTokensGetStuck	write	access only Allow Gov	No Issue
33	_wrapFTM	internal	access only Allow Gov	No Issue
34	_safeSwap	internal	Passed	No Issue

TimelockController.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	hasRole	read	Passed	No Issue
3	getRoleMemberCount	read	Passed	No Issue
4	getRoleMember	read	Passed	No Issue
5	getRoleAdmin	read	Passed	No Issue
6	grantRole	write	Passed	No Issue
7	revokeRole	write	Passed	No Issue
8	renounceRole	write	Passed	No Issue
9	_setupRole	internal	Passed	No Issue
10	_setRoleAdmin	internal	Passed	No Issue

11	_grantRole	write	Passed	No Issue
12	_revokeRole	write	Passed	No Issue
13	onlyRole	modifier	Passed	No Issue
14	receive	external	Passed	No Issue
15	isOperationPending	read	Passed	No Issue
16	isOperationReady	read	Passed	No Issue
17	isOperationDone	read	Passed	No Issue
18	getTimestamp	read	Passed	No Issue
19	getMinDelay	read	Passed	No Issue
20	hashOperation	write	Passed	No Issue
21	hashOperationBatch	write	Passed	No Issue
22	schedule	write	access only Role owner	No Issue
23	scheduleBatch	write	access only Role owner	No Issue
24	_schedule	write	Passed	No Issue
25	cancel	write	access only Role owner	No Issue
26	execute	write	access only Role owner	No Issue
27	executeBatch	write	access only Role owner	No Issue
28	_beforeCall	read	Passed	No Issue
29	_afterCall	write	Passed	No Issue
30	_call	write	Passed	No Issue
31	updateMinDelay	external	Passed	No Issue
32	updateMinDelayReduced	external	Passed	No Issue
33	setDevWalletAddress	write	Passed	No Issue
34	scheduleSet	write	access only Role owner	No Issue
35	executeSet	write	access only Role owner	No Issue
36	withdrawFTM	write	Passed	No Issue
37	withdrawBEP20	write	Passed	No Issue
38	add	write	access only Role owner	No Issue
39	earn	write	access only Role owner	No Issue
40	farm	write	access only Role owner	No Issue
41	pause	write	access only Role owner	No Issue
42	unpause	write	access only Role owner	No Issue
43	rebalance	write	access only Role owner	No Issue
44	deleverageOnce	write	access only Role owner	No Issue

45	wrapFTM	write	access only Role owner	No Issue
46	noTimeLockFunc1	write	access only Role owner	No Issue
47	noTimeLockFunc2	write	access only Role owner	No Issue
48	noTimeLockFunc3	write	access only Role owner	No Issue

xALNR.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	name	read	Passed	No Issue
3	symbol	read	Passed	No Issue
4	decimals	read	Passed	No Issue
5	totalSupply	read	Passed	No Issue
6	balanceOf	read	Passed	No Issue
7	transfer	write	Passed	No Issue
8	allowance	read	Passed	No Issue
9	approve	write	Passed	No Issue
10	transferFrom	write	Passed	No Issue
11	increaseAllowance	write	Passed	No Issue
12	decreaseAllowance	write	Passed	No Issue
13	_transfer	internal	Passed	No Issue
14	_mint	internal	Passed	No Issue
15	_burn	internal	Passed	No Issue
16	_approve	internal	Passed	No Issue
17	_setupDecimals	internal	Passed	No Issue
18	_beforeTokenTransfer	internal	Passed	No Issue
19	stakedTime	read	Passed	No Issue
20	canWithdraw	read	Passed	No Issue
21	setDelayToWithdraw	external	Passed	No Issue
22	enter	write	Critical operation lacks event log, Function input parameters lack of check	Refer Audit Findings
23	leave	write	Critical operation lacks event log, Function input parameters lack of check	Refer Audit Findings
24	ALNRBalance	external	Passed	No Issue
25	xALNRForALNR	external	Passed	No Issue
26	ALNRForxALNR	external	Passed	No Issue
27	mint	write	Passed	No Issue

28	transferFrom	write	Passed	No Issue
29	transfer	write	Passed	No Issue
30	_initDelegates	internal	Passed	No Issue
31	delegates	external	Passed	No Issue
32	delegate	external	Passed	No Issue
33	delegateBySig	external	Passed	No Issue
34	getCurrentVotes	external	Passed	No Issue
35	getPriorVotes	external	Passed	No Issue
36	_delegate	internal	Passed	No Issue
37	_moveDelegates	internal	Passed	No Issue
38	_writeCheckpoint	internal	Passed	No Issue
39	safe32	internal	Passed	No Issue
40	getChainId	internal	Passed	No Issue
41	setAdmin	write	Passed	No Issue

Nika.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	receive	external	Passed	No Issue
3	name	read	Passed	No Issue
4	symbol	read	Passed	No Issue
5	decimals	write	Passed	No Issue
6	totalSupply	read	Passed	No Issue
7	balanceOf	read	Passed	No Issue
8	allowance	read	Passed	No Issue
9	approve	write	Passed	No Issue
10	increaseAllowance	write	Passed	No Issue
11	decreaseAllowance	write	Passed	No Issue
12	transfer	write	Passed	No Issue
13	transferFrom	write	Passed	No Issue
14	openTrading	external	access only Owner	No Issue
15	_transfer	internal	Passed	No Issue
16	_executeTransfer	write	Unnecessary zero address checks	Refer Audit Findings
17	_approve	write	Passed	No Issue
18	_mint	write	Passed	No Issue
19	_burn	write	Passed	No Issue
20	swapTokensForNative	write	Passed	No Issue
21	addLiquidity	write	Passed	No Issue
22	includeToWhiteList	write	Passed	No Issue
23	_executeSwap	write	Passed	No Issue
24	excludeFromFees	write	access only Owner	No Issue
25	isExcludedFromFees	read	Passed	No Issue

26	manualSendDividend	external	Token owner can drain all dividends	Refer Audit Findings
27	excludeFromDividends	write	access only Owner	No Issue
28	isExcludedFromDividends	write	Passed	No Issue
29	setWallet	write	access only Owner	No Issue
30	setAutomatedMarketMakerPair	write	access only Owner	No Issue
31	setFee	write	Fees are freely adjustable up to over 100%	Refer Audit Findings
32	_setAutomatedMarketMakerPair	write	Passed	No Issue
33	updateUniswapV2Router	write	automatedMarketMakerPairs may get out of sync with uniswapV2Pair	Refer Audit Findings
34	claim	write	Passed	No Issue
35	compound	write	Passed	No Issue
36	withdrawableDividendOf	write	Passed	No Issue
37	withdrawnDividendOf	write	Passed	No Issue
38	accumulativeDividendOf	read	Passed	No Issue
39	getAccountInfo	read	Passed	No Issue
40	getLastClaimTime	read	Passed	No Issue
41	setSwapEnabled	external	access only Owner	No Issue
42	setTaxEnabled	external	access only Owner	No Issue
43	setCompoundingEnabled	external	access only Owner	No Issue
44	updateDividendSettings	external	access only Owner	No Issue
45	setMaxTxBPS	external	access only Owner	No Issue
46	excludeFromMaxTx	write	access only Owner	No Issue
47	isExcludedFromMaxTx	read	Passed	No Issue
48	setMaxWalletBPS	external	access only Owner	No Issue
49	excludeFromMaxWallet	write	access only Owner	No Issue
50	isExcludedFromMaxWallet	read	Passed	No Issue
51	rescueToken	external	access only Owner	No Issue
52	rescueETH	external	Owner can blacklist wallets	Refer Audit Findings
53	blackList	write	access only Owner	No Issue
54	removeFromBlacklist	write	access only Owner	No Issue
55	blackListMany	write	access only Owner	No Issue
56	unBlackListMany	write	access only Owner	No Issue

Severity Definitions

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

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

(1) Token owner can drain all dividends: **Nika.sol**
The token owner can call manualSendDividend on the token contract to transfer any or all of the dividends accumulated on the DividendTracker to any address. Users would then be unable to claim their dividends.

Resolution: We suggest removing this function. If the intent is to be able to withdraw any excess tokens accumulated on the DividendTracker, then consider modifying the function to ensure enough tokens remain for all users to be able to claim their dividends.

(2) Fees are freely adjustable up to over 100% : **Nika.sol**
The owner of the contract can set the individual fees to any variable. This might deter investors as they could be wary that these fees might one day be set to 100% to force transfers to go to the contract owner.

Resolution: We suggest adding an explicit cap to the total fee on every fee adjustment function.

Low

(1) Critical operation lacks event log:

Missing event log for:
xALNR.sol

- enter
- leave

NativeFarm.sol

- add
- set
- updatePool

Resolution: Write an event log for listed events.

(2) Function input parameters lack check:

Variable validation is not performed in the functions below:

xALNR.sol

- enter = _amount
- leave = _share

NATIVEToken.sol

- addMinter = _minter
- removeMinter = _minter

Resolution: We advise using some validation like for integer type variables should be greater than 0 and address type variables should not be address(0).

(3) maxTxBPS is initially set below the minimum enforced value: **Nika.sol**

When the contract is deployed, the antiwhale variable maxTxBPS is set to 49 (0.49%). However, in the setMaxTxBPS function that sets and enforces the bounds of this variable, the minimum value is 75 (0.75%). Users reading this function code may assume this value is the minimum the variable can be set to, when in fact it can be lower.

Resolution: Consider setting the initial value of maxTxBPS to 75 to reduce potential confusion.

(4) Owner can blacklist wallets: **Nika.sol**

Currently, the owner can blacklist wallets from purchasing or selling the tokens they hold. This might deter investors from purchasing the token.

Resolution: Consider either removing the blacklist functionality, or only allowing it to be called for a certain duration after liquidity has been added. Alternatively, consider putting the token behind a significantly long time lock so investors can see any change coming.

(5) automatedMarketMakerPairs may get out of sync with uniswapV2Pair: **Nika.sol**
Currently, the token owner needs to take care to manually call setAutomatedMarketMakerPair whenever they call updateUniswapV2Router. If they forget to do this, then the automatedMarketMakerPairs mapping will fail to contain the updated liquidity pair address.

Resolution: Add a call to _setAutomatedMarketMakerPair from updateUniswapV2Router.

Very Low / Informational / Best practices:

(1) Use latest solidity version: **xALNR.sol, Strategy.sol, NATIVEToken.sol**

Using the latest solidity will prevent any compiler level bugs.

Resolution: We suggest using version > 0.8.0.

(2) Unused function parameter / variables:
Strategy.sol
Unused function parameters _userAddress in the deposit and withdraw function.
Nika.sol

- contract ERC20 (only IERC20 is used)
- interface IERC20Metadata (only used in ERC20)
- event BlacklistEnabled (never emitted)
- library SafeMath
- variable ZERO

Resolution: We suggest removing unused function parameters and variables.

(3) Unnecessary zero address checks in _executeTransfer: **Nika.sol**

```
function _executeTransfer(
    address sender,
    address recipient,
    uint256 amount
) private {
    require(sender != address(0), "Nika: transfer from the zero address");
    require(recipient != address(0), "Nika: transfer to the zero address");
    uint256 senderBalance =  balances[sender];
```

sender and recipient are already guaranteed to be non-zero by the calling function _transfer.

Resolution: Consider removing these checks.

(4) dividendTracker should be made immutable: **Nika.sol**
Variables that are defined within the constructor but further remain unchanged should be marked as immutable to save gas and to ease the reviewing the process of third-parties.

Resolution: Consider marking this variable as immutable.

(5) _name, _symbol, DEAD and UNISWAPROUTER can be made constant: **Nika.sol**
Variables that are never modified can be indicated as such with the constant keyword. This is considered best practice since it makes the code more accessible for third-party reviewers and saves gas.

Resolution: Consider making the above variables explicitly constant.

(6) Wrong Native address: **NativeFarm.sol**

```
// Token address
address public constant NATIVE = 0xd014C8bf39119F3b68CE03EB43A199b0B9775Eb3;
// Native total supply: 17306560 = 17306560e18
```

The address specified is from the Fantom network.

Resolution: We suggest correcting the Native Token address before production.

Centralization

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

- add: The NativeFarm owner can add a new lp to the pool.
- set: The NativeFarm owner can update the given pool's NATIVE allocation point.
- setNATIVEPerSecond: The NativeFarm owner can set native token reward per second, with a cap of max native per second.
- inCaseTokensGetStuck: The NativeFarm owner can get a stuck token.
- setwithdrawalFeeFactor: The NativeFarm owner can set the withdrawal fee factor.
- addMinter: The ALNR owner can add a minter address.
- removeMinter: The ALNR owner can remove the minter address.
- deposit: The Strategy owner can receive new deposits from the user.
- withdraw: The Strategy owner can withdraw an amount from the wallet.
- openTrading: The Nika owner can set true status for open trading.
- excludeFromFees: The Nika owner can check if the account is already set to the requested state.
- manualSendDividend: The Nika owner can send a manual dividend token.
- excludeFromDividends: The Nika owner can exclude dividends.
- setWallet: The Nika owner can set the marketing Wallet address and liquidity Wallet address.
- setAutomatedMarketMakerPair: The Nika owner can set the automated market pair address.
- setFee: The Nika owner can set the _treasury Fee, _liquidityFee, _dividendFee.
- updateUniswapV2Router: The Nika owner can update the uniswap v2 router address.
- setSwapEnabled: The Nika owner can set swap enable status.
- setTaxEnabled: The Nika owner can set tax enable status.
- setCompoundingEnabled: The Nika owner can set compounding enable status.
- updateDividendSettings: The Nika owner can update the dividend setting.
- setMaxTxBPS: The Nika owner can set maximum transaction BPS.

- excludeFromMaxTx: The Nika owner can exclude the maximum transaction.
- setMaxWalletBPS: The Nika owner can set the maximum wallet BPS.
- excludeFromMaxWallet: The Nika owner can set the maximum wallet address.
- rescueToken: The Nika owner can rescue the token.
- rescueETH: The Nika owner can rescue the ETH.
- blackList: The Nika owner can add an address to the blacklist.
- removeFromBlacklist: The Nika owner can remove an address from the blacklist.
- blackListMany: The Nika owner can add multiple addresses to the blacklist.
- unBlackListMany: The Nika owner can remove multiple addresses from the blacklist.

Conclusion

We were given a contract code in the form of files. And we have used all possible tests based on given objects as files. We have observed some issues in the smart contracts, but those 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 the As-is overview section of the report.

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

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

Our Methodology

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

Manual Code Review:

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

Vulnerability Analysis:

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

Documenting Results:

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

Suggested Solutions:

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

Disclaimers

EtherAuthority.io Disclaimer

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

Due to the fact that the total number of test cases is unlimited, the audit makes no statements or warranties on the 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.

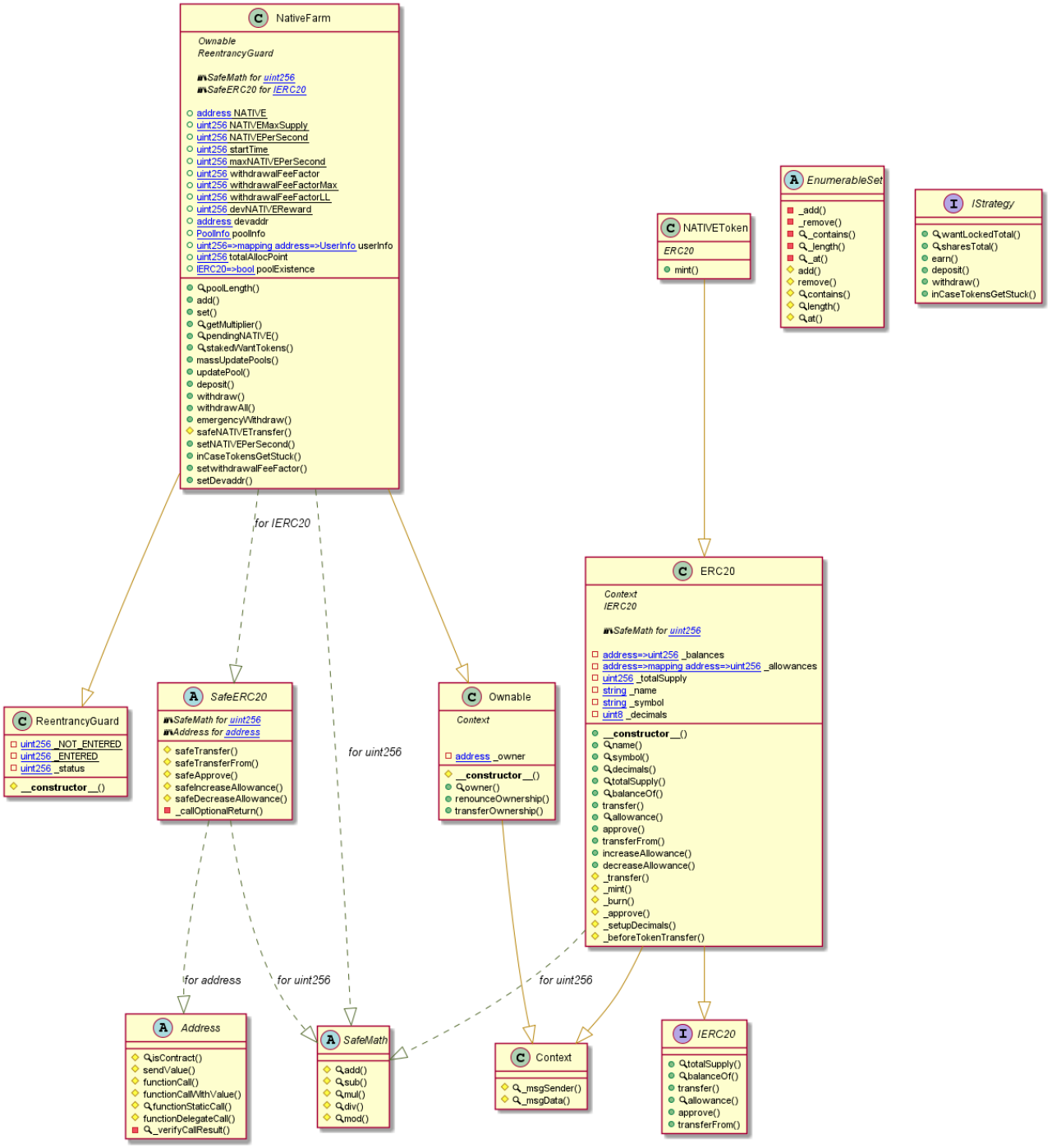
Technical Disclaimer

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.

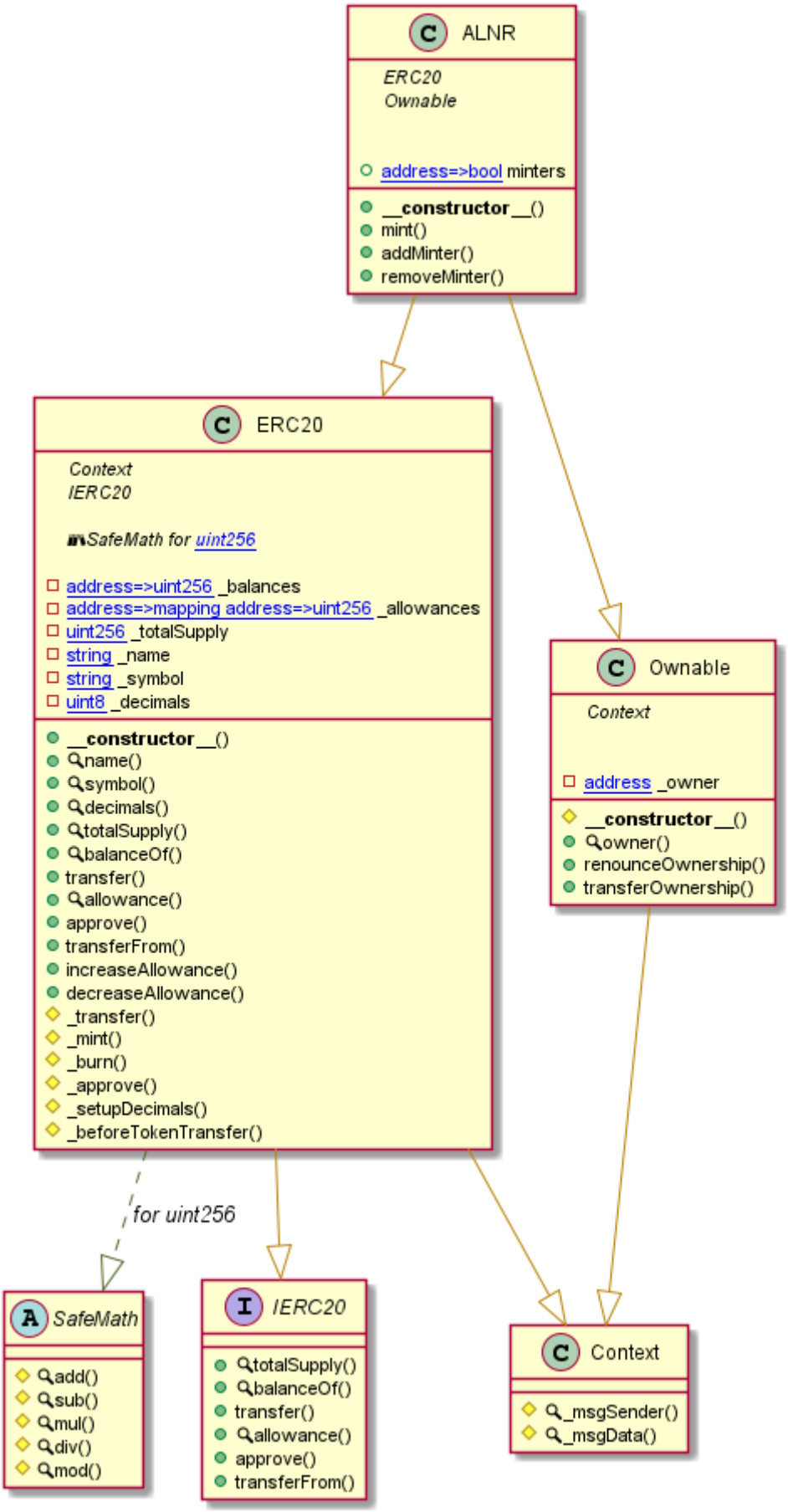
Appendix

Code Flow Diagram - Alnair Finance Protocol

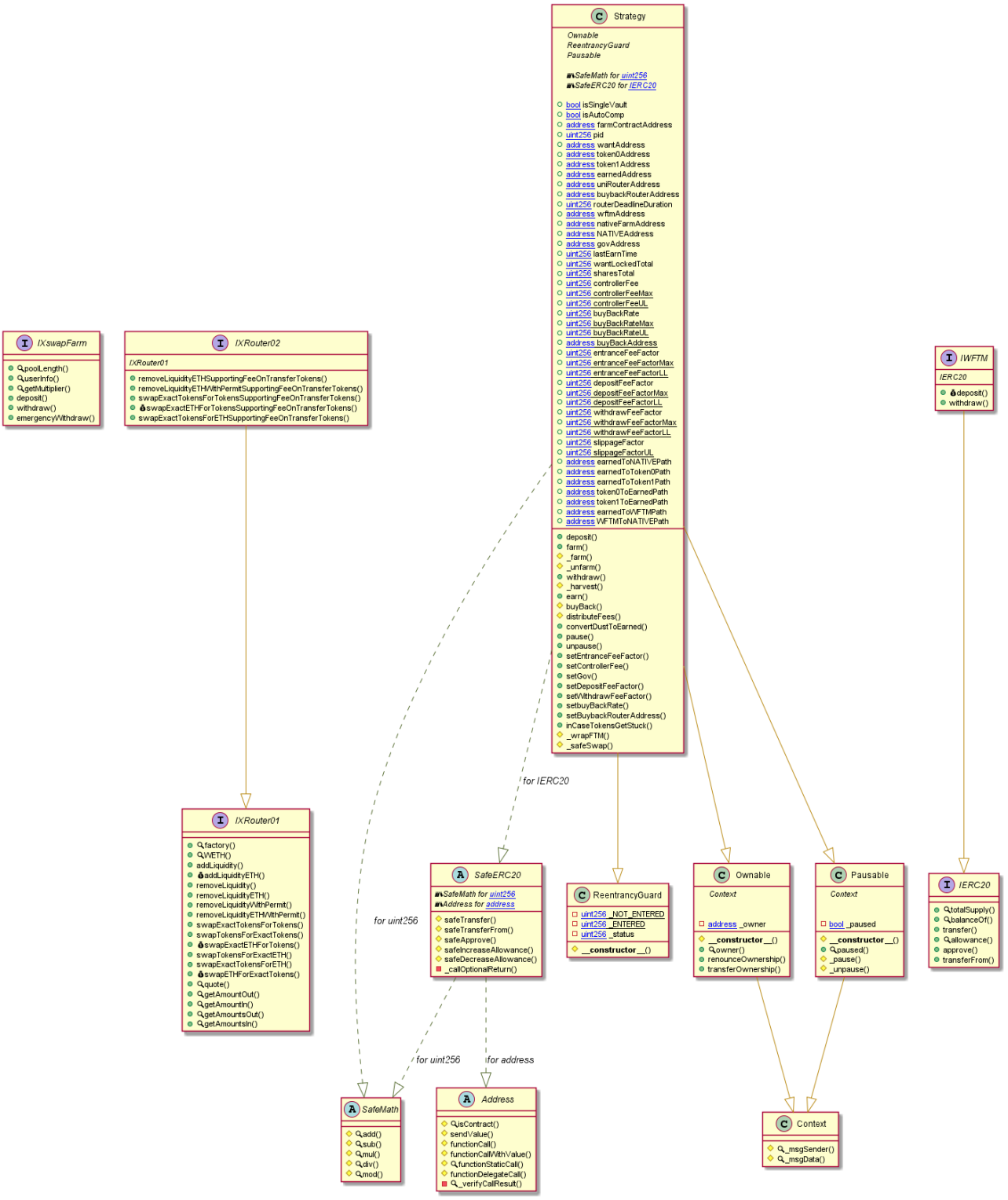
NativeFarm Diagram



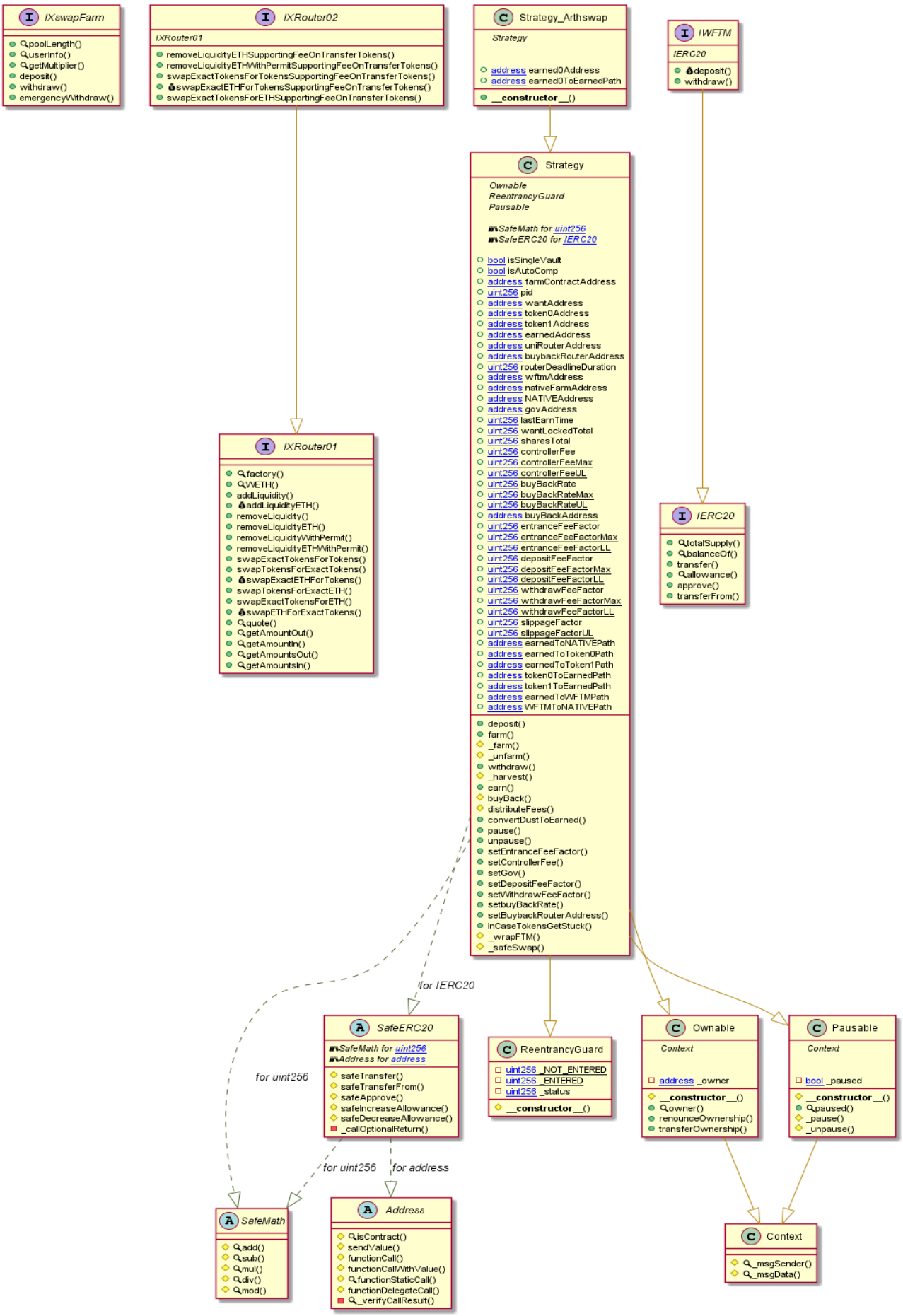
NATIVEToken Diagram



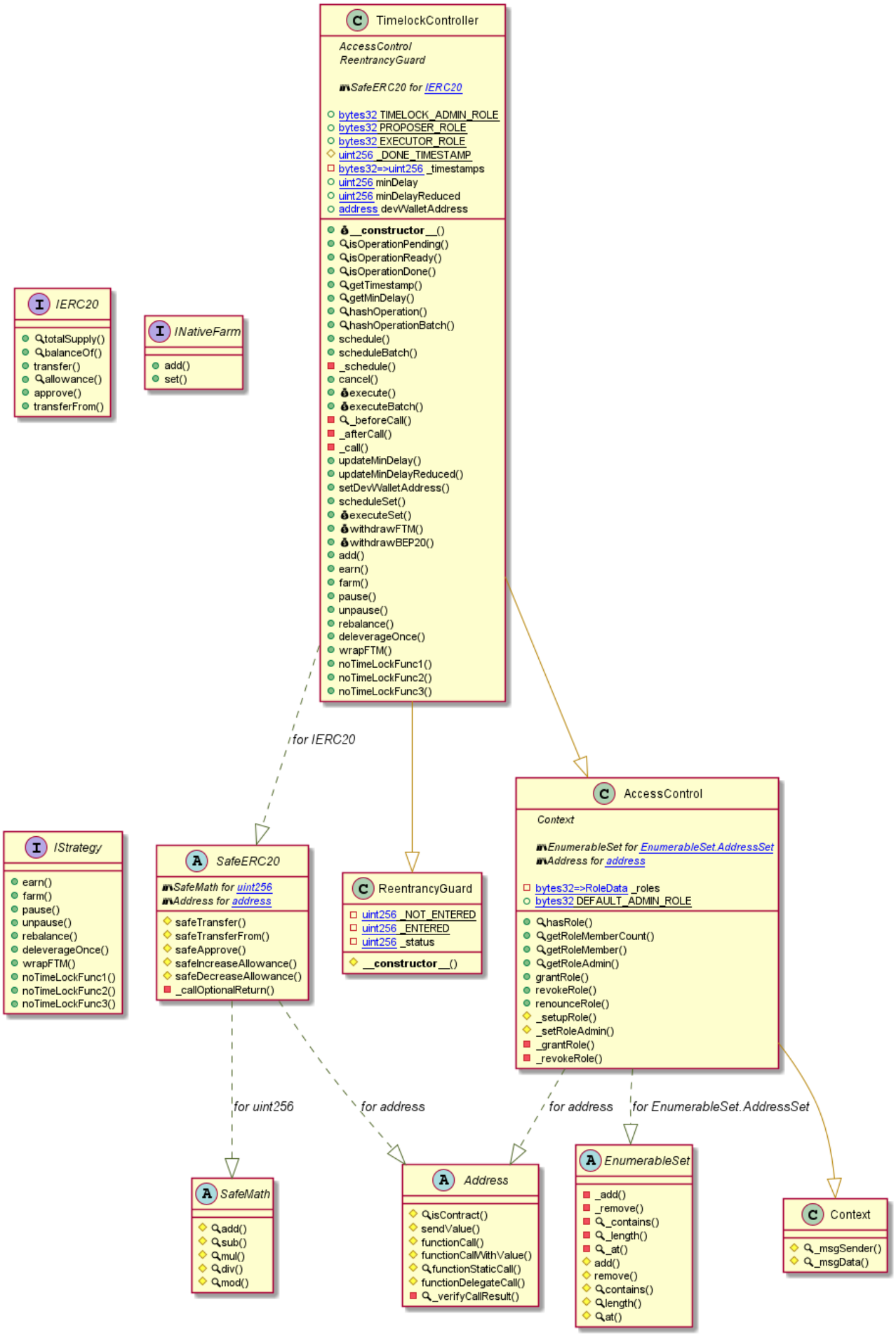
Strategy Diagram



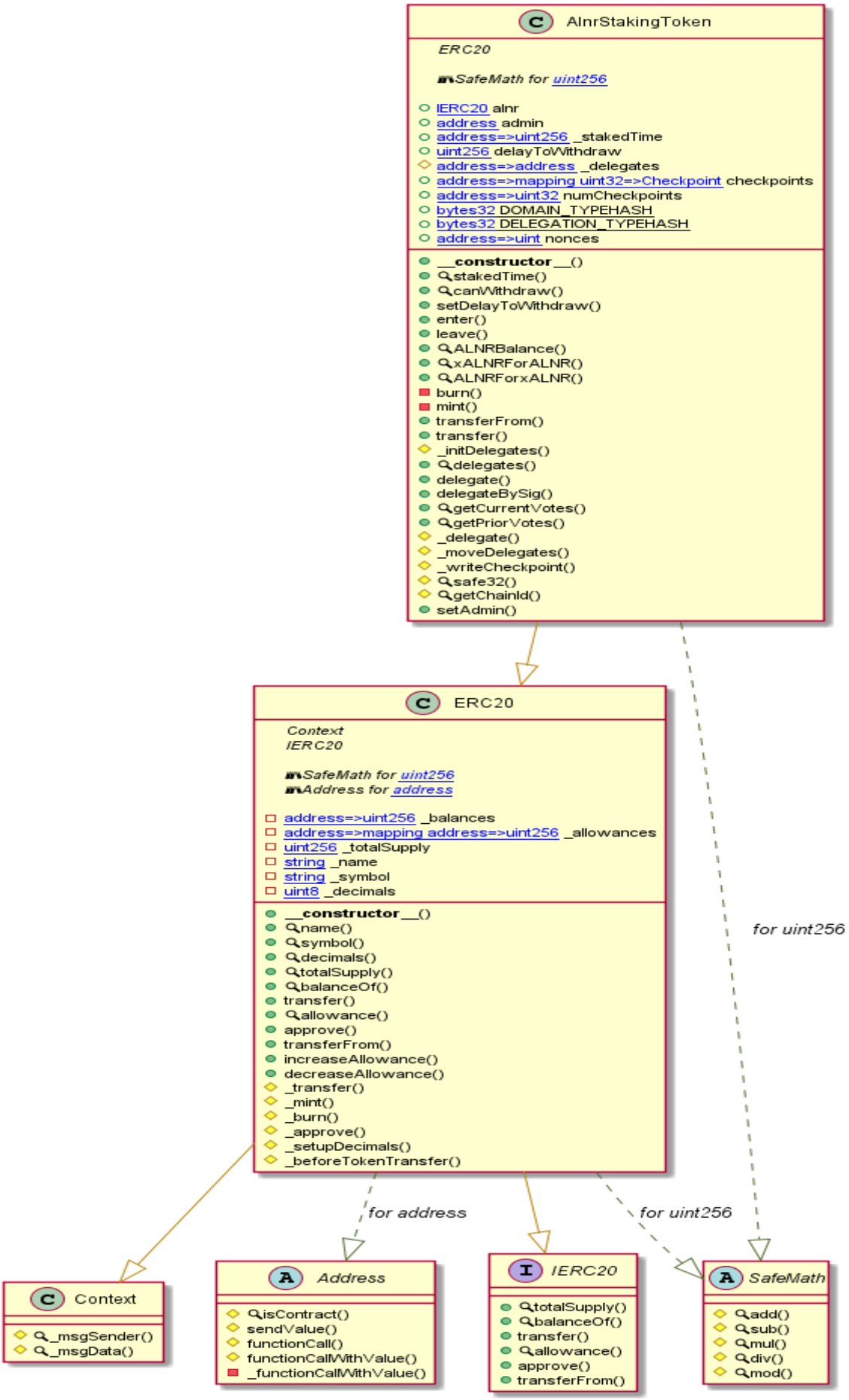
Strategy_Arthswap Diagram



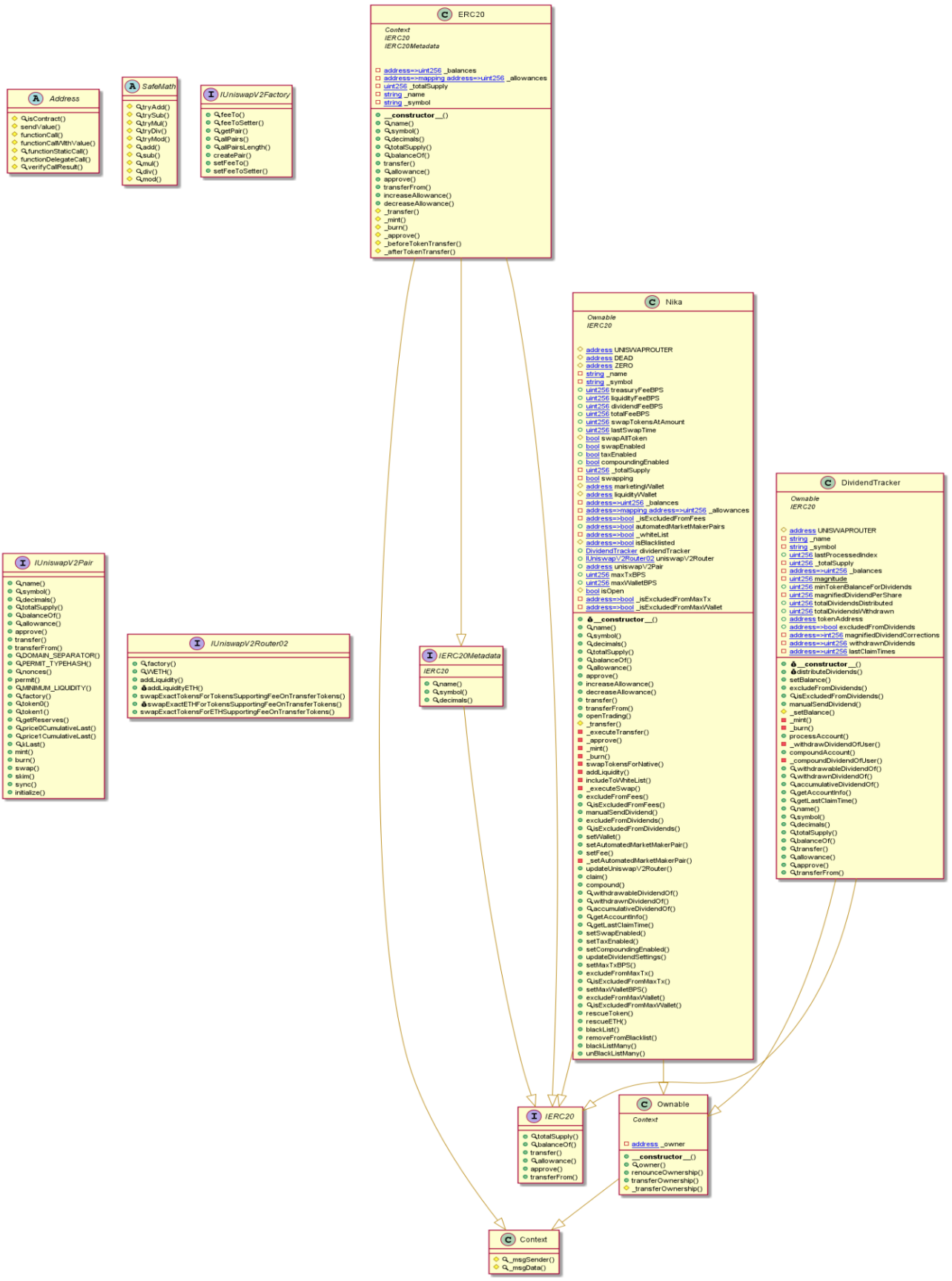
TimelockController Diagram



xALNR Diagram



Nika Diagram



Slither Results Log

Slither log >> NativeFarm.sol

```
INFO:Detectors:
NativeFarm.setDevaddr(address)._addr (NativeFarm.sol#1770) lacks a zero-check on :
                - devaddr = _addr (NativeFarm.sol#1772)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in NativeFarm.deposit(uint256,uint256) (NativeFarm.sol#1609-1637):
        External calls:
        - updatePool(_pid) (NativeFarm.sol#1610)
                - NATIVEToken(NATIVE).mint(devaddr,NATIVEReward.mul(devNATIVEReward).div(10000)) (NativeFarm.sol#1595-1598)
                - NATIVEToken(NATIVE).mint(address(this),NATIVEReward) (NativeFarm.sol#1600)
        - safeNATIVETransfer(msg.sender,pending) (NativeFarm.sol#1620)
                - IERC20(NATIVE).transfer(_to,NATIVEBal) (NativeFarm.sol#1732)
                - IERC20(NATIVE).transfer(_to,_NATIVEAmt) (NativeFarm.sol#1734)
        - pool.want.safeTransferFrom(address(msg.sender),address(this),_wantAmt) (NativeFarm.sol#1624-1628)
        - pool.want.safeIncreaseAllowance(pool.strat,_wantAmt) (NativeFarm.sol#1630)
        - sharesAdded = IStrategy(poolInfo[_pid].strat).deposit(msg.sender,_wantAmt) (NativeFarm.sol#1631-1632)
        Event emitted after the call(s):
        - Deposit(msg.sender,_pid,_wantAmt) (NativeFarm.sol#1636)
Reentrancy in NativeFarm.emergencyWithdraw(uint256) (NativeFarm.sol#1702-1726):
        External calls:
        - IStrategy(poolInfo[_pid].strat).withdraw(msg.sender,amount) (NativeFarm.sol#1711)
        - pool.want.safeTransfer(address(msg.sender),amountToSend) (NativeFarm.sol#1718)
        - pool.want.safeTransfer(owner(),fee) (NativeFarm.sol#1719)
        - pool.want.safeTransfer(address(msg.sender),amount) (NativeFarm.sol#1721)
        Event emitted after the call(s):
        - EmergencyWithdraw(msg.sender,_pid,amount) (NativeFarm.sol#1723)
Reentrancy in NativeFarm.withdraw(uint256,uint256) (NativeFarm.sol#1640-1695):
        External calls:
        - updatePool(_pid) (NativeFarm.sol#1641)
                - NATIVEToken(NATIVE).mint(devaddr,NATIVEReward.mul(devNATIVEReward).div(10000)) (NativeFarm.sol#1595-1598)
                - NATIVEToken(NATIVE).mint(address(this),NATIVEReward) (NativeFarm.sol#1600)
        - safeNATIVETransfer(msg.sender,pending) (NativeFarm.sol#1659)
                - IERC20(NATIVE).transfer(_to,NATIVEBal) (NativeFarm.sol#1732)
                - IERC20(NATIVE).transfer(_to,_NATIVEAmt) (NativeFarm.sol#1734)
        - sharesRemoved = IStrategy(poolInfo[_pid].strat).withdraw(msg.sender,_wantAmt) (NativeFarm.sol#1668-1669)
```

```
        - pool.want.safeTransfer(address(msg.sender),_wantAmtToSend) (NativeFarm.sol#1687)
        - pool.want.safeTransfer(owner(),fee) (NativeFarm.sol#1688)
        - pool.want.safeTransfer(address(msg.sender),_wantAmt) (NativeFarm.sol#1690)
        Event emitted after the call(s):
        - Withdraw(msg.sender,_pid,_wantAmt) (NativeFarm.sol#1694)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
NativeFarm.add(uint256,IERC20,address) (NativeFarm.sol#1484-1501) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > startTime (NativeFarm.sol#1489)
NativeFarm.pendingNATIVE(uint256,address) (NativeFarm.sol#1527-1547) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > pool.lastRewardTime && sharesTotal != 0 (NativeFarm.sol#1536)
NativeFarm.massUpdatePools() (NativeFarm.sol#1568-1573) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (NativeFarm.sol#1570)
NativeFarm.updatePool(uint256) (NativeFarm.sol#1576-1606) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= pool.lastRewardTime (NativeFarm.sol#1578)
        - multiplier <= 0 (NativeFarm.sol#1587)
NativeFarm.withdraw(uint256,uint256) (NativeFarm.sol#1640-1695) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(user.shares > 0,user.shares is 0) (NativeFarm.sol#1650)
        - require(bool,string)(sharesTotal > 0,sharesTotal is 0) (NativeFarm.sol#1651)
        - _wantAmt > amount (NativeFarm.sol#1664)
        - _wantAmt > 0 (NativeFarm.sol#1667)
        - sharesRemoved > user.shares (NativeFarm.sol#1671)
        - wantBal < _wantAmt (NativeFarm.sol#1678)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (NativeFarm.sol#612-623) uses assembly
        - INLINE ASM (NativeFarm.sol#619-621)
Address._verifyCallResult(bool,bytes,string) (NativeFarm.sol#816-837) uses assembly
        - INLINE ASM (NativeFarm.sol#829-832)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

```
        -require(bool,string)(poolExistence[_want] == false,nonDuplicated: Duplicated LPToken) (NativeFarm.sol#1479)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
INFO:Detectors:
Address.functionCall(address,bytes) (NativeFarm.sol#673-678) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (NativeFarm.sol#705-717) is never used and should be removed
Address.functionDelegateCall(address,bytes) (NativeFarm.sol#786-796) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (NativeFarm.sol#804-814) is never used and should be removed
Address.functionStaticCall(address,bytes) (NativeFarm.sol#749-760) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (NativeFarm.sol#768-778) is never used and should be removed
Address.sendValue(address,uint256) (NativeFarm.sol#641-653) is never used and should be removed
Context._msgData() (NativeFarm.sol#10-13) is never used and should be removed
ERC20._burn(address,uint256) (NativeFarm.sol#524-535) is never used and should be removed
ERC20._mint(address,uint256) (NativeFarm.sol#503-511) is never used and should be removed
ERC20._setupDecimals(uint8) (NativeFarm.sol#569-571) is never used and should be removed
EnumerableSet._add(EnumerableSet.Set,bytes32) (NativeFarm.sol#981-991) is never used and should be removed
EnumerableSet._at(EnumerableSet.Set,uint256) (NativeFarm.sol#1062-1072) is never used and should be removed
EnumerableSet._contains(EnumerableSet.Set,bytes32) (NativeFarm.sol#1037-1043) is never used and should be removed
EnumerableSet._length(EnumerableSet.Set) (NativeFarm.sol#1048-1050) is never used and should be removed
EnumerableSet._remove(EnumerableSet.Set,bytes32) (NativeFarm.sol#999-1032) is never used and should be removed
EnumerableSet.add(EnumerableSet.AddressSet,address) (NativeFarm.sol#1154-1159) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (NativeFarm.sol#1086-1091) is never used and should be removed
EnumerableSet.add(EnumerableSet.UintSet,uint256) (NativeFarm.sol#1222-1224) is never used and should be removed
EnumerableSet.at(EnumerableSet.AddressSet,uint256) (NativeFarm.sol#1202-1208) is never used and should be removed
EnumerableSet.at(EnumerableSet.Bytes32Set,uint256) (NativeFarm.sol#1134-1140) is never used and should be removed
EnumerableSet.at(EnumerableSet.UintSet,uint256) (NativeFarm.sol#1267-1273) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet,address) (NativeFarm.sol#1177-1183) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (NativeFarm.sol#1109-1115) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (NativeFarm.sol#1242-1248) is never used and should be removed
EnumerableSet.length(EnumerableSet.AddressSet) (NativeFarm.sol#1188-1190) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (NativeFarm.sol#1120-1122) is never used and should be removed
EnumerableSet.length(EnumerableSet.UintSet) (NativeFarm.sol#1253-1255) is never used and should be removed
EnumerableSet.remove(EnumerableSet.AddressSet,address) (NativeFarm.sol#1167-1172) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (NativeFarm.sol#1099-1104) is never used and should be removed
EnumerableSet.remove(EnumerableSet.UintSet,uint256) (NativeFarm.sol#1232-1237) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (NativeFarm.sol#874-891) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (NativeFarm.sol#910-928) is never used and should be removed
SafeMath.mod(uint256,uint256) (NativeFarm.sol#145-147) is never used and should be removed
```

```
SafeMath.mod(uint256,uint256,string) (NativeFarm.sol#161-168) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (NativeFarm.sol#641-653):
        - (success) = recipient.call{value: amount}() (NativeFarm.sol#648)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (NativeFarm.sol#725-741):
        - (success,returndata) = target.call{value: value}(data) (NativeFarm.sol#738-739)
Low level call in Address.functionStaticCall(address,bytes,string) (NativeFarm.sol#768-778):
        - (success,returndata) = target.staticcall(data) (NativeFarm.sol#776)
Low level call in Address.functionDelegateCall(address,bytes,string) (NativeFarm.sol#804-814):
        - (success,returndata) = target.delegatecall(data) (NativeFarm.sol#812)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter NativeFarm.add(uint256,IERC20,address)._allocPoint (NativeFarm.sol#1485) is not in mixedCase
Parameter NativeFarm.add(uint256,IERC20,address)._want (NativeFarm.sol#1486) is not in mixedCase
```

```
Constant NativeFarm.NATIVEMaxSupply (NativeFarm.sol#1443) is not in UPPER_CASE_WITH_UNDERSCORES
Constant NativeFarm.NATIVEPerSecond (NativeFarm.sol#1445) is not in UPPER_CASE_WITH_UNDERSCORES
Constant NativeFarm.startTime (NativeFarm.sol#1447) is not in UPPER_CASE_WITH_UNDERSCORES
Constant NativeFarm.maxNATIVEPerSecond (NativeFarm.sol#1449) is not in UPPER_CASE_WITH_UNDERSCORES
Constant NativeFarm.withdrawalFeeFactorMax (NativeFarm.sol#1453) is not in UPPER_CASE_WITH_UNDERSCORES
Constant NativeFarm.withdrawalFeeFactorLL (NativeFarm.sol#1454) is not in UPPER_CASE_WITH_UNDERSCORES
Constant NativeFarm.devNATIVEReward (NativeFarm.sol#1456) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (NativeFarm.sol#11)" inContext (NativeFarm.sol#5-14)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
NATIVEToken (NativeFarm.sol#1378-1380) does not implement functions:
        - NATIVEToken.mint(address,uint256) (NativeFarm.sol#1379)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (NativeFarm.sol#286-288)
symbol() should be declared external:
        - ERC20.symbol() (NativeFarm.sol#294-296)
decimals() should be declared external:
        - ERC20.decimals() (NativeFarm.sol#311-313)
totalSupply() should be declared external:
        - ERC20.totalSupply() (NativeFarm.sol#318-320)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (NativeFarm.sol#325-327)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (NativeFarm.sol#337-345)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (NativeFarm.sol#350-358)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (NativeFarm.sol#367-375)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (NativeFarm.sol#390-405)
increaseAllowance(address,uint256) should be declared external:
```

```
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (NativeFarm.sol#446-460)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (NativeFarm.sol#1315-1318)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (NativeFarm.sol#1324-1331)
setDevaddr(address) should be declared external:
        - NativeFarm.setDevaddr(address) (NativeFarm.sol#1770-1773)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:NativeFarm.sol analyzed (12 contracts with 75 detectors), 113 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> NATIVEToken.sol

```
INFO:Detectors:
ALNR.constructor(string,string,address)._name (NATIVEToken.sol#653) shadows:
        - ERC20._name (NATIVEToken.sol#264) (state variable)
ALNR.constructor(string,string,address)._symbol (NATIVEToken.sol#653) shadows:
        - ERC20._symbol (NATIVEToken.sol#265) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Context._msgData() (NATIVEToken.sol#10-13) is never used and should be removed
ERC20._burn(address,uint256) (NATIVEToken.sol#521-532) is never used and should be removed
ERC20._setupDecimals(uint8) (NATIVEToken.sol#566-568) is never used and should be removed
SafeMath.div(uint256,uint256) (NATIVEToken.sol#105-107) is never used and should be removed
SafeMath.div(uint256,uint256,string) (NATIVEToken.sol#121-131) is never used and should be removed
SafeMath.mod(uint256,uint256) (NATIVEToken.sol#145-147) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (NATIVEToken.sol#161-168) is never used and should be removed
SafeMath.mul(uint256,uint256) (NATIVEToken.sol#79-91) is never used and should be removed
SafeMath.sub(uint256,uint256) (NATIVEToken.sol#44-46) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Parameter ALNR.mint(address,uint256)._to (NATIVEToken.sol#658) is not in mixedCase
Parameter ALNR.mint(address,uint256)._amount (NATIVEToken.sol#658) is not in mixedCase
Parameter ALNR.addMinter(address)._minter (NATIVEToken.sol#663) is not in mixedCase
Parameter ALNR.removeMinter(address)._minter (NATIVEToken.sol#667) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (NATIVEToken.sol#11)" inContext (NATIVEToken.sol#5-14)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
ALNR.constructor(string,string,address) (NATIVEToken.sol#653-656) uses literals with too many digits:
        - _mint(msg.sender,6_800_000_000000_000000_000000) (NATIVEToken.sol#654)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
name() should be declared external:
        - ERC20.name() (NATIVEToken.sol#286-288)
symbol() should be declared external:
        - ERC20.symbol() (NATIVEToken.sol#294-296)
```

```
decimals() should be declared external:
        - ERC20.decimals() (NATIVEToken.sol#311-313)
totalSupply() should be declared external:
        - ERC20.totalSupply() (NATIVEToken.sol#318-320)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (NATIVEToken.sol#325-327)
```

```
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (NATIVEToken.sol#337-345)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (NATIVEToken.sol#350-358)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (NATIVEToken.sol#367-375)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (NATIVEToken.sol#390-405)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (NATIVEToken.sol#419-430)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (NATIVEToken.sol#446-460)
owner() should be declared external:
        - Ownable.owner() (NATIVEToken.sol#611-613)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (NATIVEToken.sol#630-633)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (NATIVEToken.sol#639-646)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:NATIVEToken.sol analyzed (6 contracts with 75 detectors), 31 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Strategy.sol

```
INFO:Detectors:
Strategy.setGov(address) (Strategy.sol#1433-1435) should emit an event for:
        - govAddress = _govAddress (Strategy.sol#1434)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
INFO:Detectors:
Strategy.setEntranceFeeFactor(uint256) (Strategy.sol#1422-1426) should emit an event for:
        - entranceFeeFactor = _entranceFeeFactor (Strategy.sol#1425)
Strategy.setControllerFee(uint256) (Strategy.sol#1428-1431) should emit an event for:
        - controllerFee = _controllerFee (Strategy.sol#1430)
Strategy.setDepositFeeFactor(uint256) (Strategy.sol#1437-1441) should emit an event for:
        - depositFeeFactor = _depositFeeFactor (Strategy.sol#1440)
Strategy.setWithdrawFeeFactor(uint256) (Strategy.sol#1443-1447) should emit an event for:
        - withdrawFeeFactor = _withdrawFeeFactor (Strategy.sol#1446)
Strategy.setbuyBackRate(uint256) (Strategy.sol#1449-1452) should emit an event for:
        - buyBackRate = _buyBackRate (Strategy.sol#1451)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Strategy.setGov(address)._govAddress (Strategy.sol#1433) lacks a zero-check on :
                - govAddress = _govAddress (Strategy.sol#1434)
Strategy.setBuybackRouterAddress(address)._buybackRouterAddress (Strategy.sol#1454) lacks a zero-check on :
                - buybackRouterAddress = _buybackRouterAddress (Strategy.sol#1455)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in Strategy.deposit(address,uint256) (Strategy.sol#1099-1135):
        External calls:
        - IERC20(wantAddress).safeTransferFrom(address(msg.sender),address(this),_wantAmt) (Strategy.sol#1107-1111)
        State variables written after the call(s):
        - sharesTotal = sharesTotal.add(sharesAdded) (Strategy.sol#1126)
        - wantLockedTotal = wantLockedTotal.add(_wantAmt) (Strategy.sol#1131)
Reentrancy in Strategy.earn() (Strategy.sol#1200-1290):
        External calls:
        - _harvest() (Strategy.sol#1204)
                - IXswapFarm(farmContractAddress).withdraw(pid,_wantAmt) (Strategy.sol#1151)
        - earnedAmt = distributeFees(earnedAmt) (Strategy.sol#1209)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Strategy.sol#824-828)
                - IERC20(earnedAddress).safeTransfer(govAddress,fee) (Strategy.sol#1361)
                - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
```

```
        - earnedAmt = buyBack(earnedAmt) (Strategy.sol#1210)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Strategy.sol#824-828)
                - IXRouter02(_uniRouterAddress).swapExactTokensForTokensSupportingFeeOnTransferTokens(_amountIn,amountOut.mul(_slippageF
actor).div(1000),_path,_to,_deadline) (Strategy.sol#1488-1495)
                - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
                - IERC20(earnedAddress).safeIncreaseAllowance(uniRouterAddress,buyBackAmt) (Strategy.sol#1304-1307)
                - IXRouter02(uniRouterAddress).swapExactTokensForTokensSupportingFeeOnTransferTokens(buyBackAmt,0,earnedToWFTMPath,addre
ss(this),now + routerDeadlineDuration) (Strategy.sol#1309-1316)
                - IERC20(wftmAddress).safeIncreaseAllowance(buybackRouterAddress,wftmAmt) (Strategy.sol#1322-1325)
                - IXRouter02(buybackRouterAddress).swapExactTokensForTokensSupportingFeeOnTransferTokens(wftmAmt,0,WFTMToNATIVEPath,buyB
ackAddress,now + routerDeadlineDuration) (Strategy.sol#1327-1334)
                - IERC20(earnedAddress).safeIncreaseAllowance(uniRouterAddress,buyBackAmt) (Strategy.sol#1337-1340)
        - IERC20(earnedAddress).safeIncreaseAllowance(uniRouterAddress,earnedAmt) (Strategy.sol#1214-1217)
        - _safeSwap(uniRouterAddress,earnedAmt,slippageFactor,earnedToToken0Path,address(this),now + routerDeadlineDuration) (Strategy.s
ol#1220-1227)
                - IXRouter02(_uniRouterAddress).swapExactTokensForTokensSupportingFeeOnTransferTokens(_amountIn,amountOut.mul(_slippageF
actor).div(1000),_path,_to,_deadline) (Strategy.sol#1488-1495)
        External calls sending eth:
        - earnedAmt = distributeFees(earnedAmt) (Strategy.sol#1209)
                - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
        - earnedAmt = buyBack(earnedAmt) (Strategy.sol#1210)
                - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
        State variables written after the call(s):
        - lastEarnTime = block.timestamp (Strategy.sol#1229)
Reentrancy in Strategy.earn() (Strategy.sol#1200-1290):
        External calls:
        - _harvest() (Strategy.sol#1204)
                - IXswapFarm(farmContractAddress).withdraw(pid,_wantAmt) (Strategy.sol#1151)
        - earnedAmt = distributeFees(earnedAmt) (Strategy.sol#1209)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Strategy.sol#824-828)
                - IERC20(earnedAddress).safeTransfer(govAddress,fee) (Strategy.sol#1361)
                - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
        - earnedAmt = buyBack(earnedAmt) (Strategy.sol#1210)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Strategy.sol#824-828)
                - IXRouter02(_uniRouterAddress).swapExactTokensForTokensSupportingFeeOnTransferTokens(_amountIn,amountOut.mul(_slippageF
actor).div(1000),_path,_to,_deadline) (Strategy.sol#1488-1495)
                - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Address.isContract(address) (Strategy.sol#411-422) uses assembly
        - INLINE ASM (Strategy.sol#418-420)
Address._verifyCallResult(bool,bytes,string) (Strategy.sol#615-636) uses assembly
        - INLINE ASM (Strategy.sol#628-631)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

```
INFO:Detectors:
Address.functionCall(address,bytes) (Strategy.sol#472-477) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Strategy.sol#504-516) is never used and should be removed
Address.functionDelegateCall(address,bytes) (Strategy.sol#585-595) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (Strategy.sol#603-613) is never used and should be removed
Address.functionStaticCall(address,bytes) (Strategy.sol#548-559) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (Strategy.sol#567-577) is never used and should be removed
Address.sendValue(address,uint256) (Strategy.sol#440-452) is never used and should be removed
Context._msgData() (Strategy.sol#894-897) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (Strategy.sol#757-774) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (Strategy.sol#793-811) is never used and should be removed
SafeMath.mod(uint256,uint256) (Strategy.sol#367-369) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Strategy.sol#383-390) is never used and should be removed
Strategy._wrapFTM() (Strategy.sol#1468-1474) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (Strategy.sol#440-452):
        - (success) = recipient.call{value: amount}() (Strategy.sol#447)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (Strategy.sol#524-540):
        - (success,returndata) = target.call{value: value}(data) (Strategy.sol#537-538)
Low level call in Address.functionStaticCall(address,bytes,string) (Strategy.sol#567-577):
        - (success,returndata) = target.staticcall(data) (Strategy.sol#575)
Low level call in Address.functionDelegateCall(address,bytes,string) (Strategy.sol#603-613):
        - (success,returndata) = target.delegatecall(data) (Strategy.sol#611)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IXRouter01.WETH() (Strategy.sol#36) is not in mixedCase
Parameter Strategy.deposit(address,uint256)._wantAmt (Strategy.sol#1099) is not in mixedCase
Parameter Strategy.withdraw(address,uint256)._wantAmt (Strategy.sol#1154) is not in mixedCase
Parameter Strategy.buyBack(uint256)._earnedAmt (Strategy.sol#1292) is not in mixedCase
```

```
Constant Strategy.slippageFactorUL (Strategy.sol#1083) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Strategy.WFTMToNATIVEPath (Strategy.sol#1091) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (Strategy.sol#895)" inContext (Strategy.sol#889-898)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IXRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Strategy.sol#41) is to
o similar to IXRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Strategy.sol#42)
Variable Strategy.earnedToToken0Path (Strategy.sol#1086) is too similar to Strategy.earnedToToken1Path (Strategy.sol#1087)
Variable Strategy.token0Address (Strategy.sol#1045) is too similar to Strategy.token1Address (Strategy.sol#1046)
Variable Strategy.token0ToEarnedPath (Strategy.sol#1088) is too similar to Strategy.token1ToEarnedPath (Strategy.sol#1089)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
Strategy.slitherConstructorConstantVariables() (Strategy.sol#1033-1497) uses literals with too many digits:
        - buyBackAddress = 0x000000000000000000000000000000000000dEaD (Strategy.sol#1068)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Strategy.NATIVEAddress (Strategy.sol#1054) should be constant
Strategy.earnedAddress (Strategy.sol#1047) should be constant
Strategy.farmContractAddress (Strategy.sol#1042) should be constant
Strategy.isAutoComp (Strategy.sol#1040) should be constant
Strategy.isSingleVault (Strategy.sol#1039) should be constant
Strategy.nativeFarmAddress (Strategy.sol#1053) should be constant
Strategy.pid (Strategy.sol#1043) should be constant
Strategy.routerDeadlineDuration (Strategy.sol#1050) should be constant
Strategy.slippageFactor (Strategy.sol#1082) should be constant
Strategy.token0Address (Strategy.sol#1045) should be constant
Strategy.token1Address (Strategy.sol#1046) should be constant
Strategy.uniRouterAddress (Strategy.sol#1048) should be constant
Strategy.wantAddress (Strategy.sol#1044) should be constant
Strategy.wftmAddress (Strategy.sol#1052) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

```
INFO:Detectors:
owner() should be declared external:
        - Ownable.owner() (Strategy.sol#920-922)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Strategy.sol#939-942)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Strategy.sol#948-955)
paused() should be declared external:
        - Pausable.paused() (Strategy.sol#981-983)
farm() should be declared external:
        - Strategy.farm() (Strategy.sol#1137-1139)
earn() should be declared external:
        - Strategy.earn() (Strategy.sol#1200-1290)
convertDustToEarned() should be declared external:
        - Strategy.convertDustToEarned() (Strategy.sol#1369-1412)
pause() should be declared external:
        - Strategy.pause() (Strategy.sol#1414-1416)
setEntranceFeeFactor(uint256) should be declared external:
        - Strategy.setEntranceFeeFactor(uint256) (Strategy.sol#1422-1426)
setControllerFee(uint256) should be declared external:
        - Strategy.setControllerFee(uint256) (Strategy.sol#1428-1431)
setGov(address) should be declared external:
        - Strategy.setGov(address) (Strategy.sol#1433-1435)
setDepositFeeFactor(uint256) should be declared external:
        - Strategy.setDepositFeeFactor(uint256) (Strategy.sol#1437-1441)
setWithdrawFeeFactor(uint256) should be declared external:
        - Strategy.setWithdrawFeeFactor(uint256) (Strategy.sol#1443-1447)
setbuyBackRate(uint256) should be declared external:
        - Strategy.setbuyBackRate(uint256) (Strategy.sol#1449-1452)
setBuybackRouterAddress(address) should be declared external:
        - Strategy.setBuybackRouterAddress(address) (Strategy.sol#1454-1456)
inCaseTokensGetStuck(address,uint256,address) should be declared external:
        - Strategy.inCaseTokensGetStuck(address,uint256,address) (Strategy.sol#1458-1466)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Strategy.sol analyzed (13 contracts with 75 detectors), 116 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Strategy_Arthswap.sol

```
INFO:Detectors:
Strategy.setGov(address) (Strategy_Arthswap.sol#1432-1434) should emit an event for:
        - govAddress = _govAddress (Strategy_Arthswap.sol#1433)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control
```

```
INFO:Detectors:
Strategy.setGov(address)._govAddress (Strategy_Arthswap.sol#1432) lacks a zero-check on :
                - govAddress = _govAddress (Strategy_Arthswap.sol#1433)
Strategy.setBuybackRouterAddress(address)._buybackRouterAddress (Strategy_Arthswap.sol#1453) lacks a zero-check on :
                - buybackRouterAddress = _buybackRouterAddress (Strategy_Arthswap.sol#1454)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in Strategy.deposit(address,uint256) (Strategy_Arthswap.sol#1098-1134):
        External calls:
        - IERC20(wantAddress).safeTransferFrom(address(msg.sender),address(this),_wantAmt) (Strategy_Arthswap.sol#1106-1110)
        State variables written after the call(s):
        - sharesTotal = sharesTotal.add(sharesAdded) (Strategy_Arthswap.sol#1125)
        - wantLockedTotal = wantLockedTotal.add(_wantAmt) (Strategy_Arthswap.sol#1130)
Reentrancy in Strategy.earn() (Strategy_Arthswap.sol#1199-1289):
        External calls:
        - _harvest() (Strategy_Arthswap.sol#1203)
                - IXswapFarm(farmContractAddress).withdraw(pid,_wantAmt) (Strategy_Arthswap.sol#1150)
        - earnedAmt = distributeFees(earnedAmt) (Strategy_Arthswap.sol#1208)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Strategy_Arthswap.sol#823-827)
                - IERC20(earnedAddress).safeTransfer(govAddress,fee) (Strategy_Arthswap.sol#1360)
                - (success,returndata) = target.call{value: value}(data) (Strategy_Arthswap.sol#536-537)
        - earnedAmt = buyBack(earnedAmt) (Strategy_Arthswap.sol#1209)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (Strategy_Arthswap.sol#823-827)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Address.isContract(address) (Strategy_Arthswap.sol#410-421) uses assembly
        - INLINE ASM (Strategy_Arthswap.sol#417-419)
Address._verifyCallResult(bool,bytes,string) (Strategy_Arthswap.sol#614-635) uses assembly
        - INLINE ASM (Strategy_Arthswap.sol#627-630)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (Strategy_Arthswap.sol#471-476) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Strategy_Arthswap.sol#503-515) is never used and should be removed
Address.functionDelegateCall(address,bytes) (Strategy_Arthswap.sol#584-594) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (Strategy_Arthswap.sol#602-612) is never used and should be removed
Address.functionStaticCall(address,bytes) (Strategy_Arthswap.sol#547-558) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (Strategy_Arthswap.sol#566-576) is never used and should be removed
Address.sendValue(address,uint256) (Strategy_Arthswap.sol#439-451) is never used and should be removed
Context._msgData() (Strategy_Arthswap.sol#893-896) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (Strategy_Arthswap.sol#756-773) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (Strategy_Arthswap.sol#792-810) is never used and should be removed
SafeMath.mod(uint256,uint256) (Strategy_Arthswap.sol#366-368) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Strategy_Arthswap.sol#382-389) is never used and should be removed
Strategy._wrapFTM() (Strategy_Arthswap.sol#1467-1473) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (Strategy_Arthswap.sol#439-451):
        - (success) = recipient.call{value: amount}() (Strategy_Arthswap.sol#446)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (Strategy_Arthswap.sol#523-539):
        - (success,returndata) = target.call{value: value}(data) (Strategy_Arthswap.sol#536-537)
Low level call in Address.functionStaticCall(address,bytes,string) (Strategy_Arthswap.sol#566-576):
        - (success,returndata) = target.staticcall(data) (Strategy_Arthswap.sol#574)
Low level call in Address.functionDelegateCall(address,bytes,string) (Strategy_Arthswap.sol#602-612):
        - (success,returndata) = target.delegatecall(data) (Strategy_Arthswap.sol#610)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

```
INFO:Detectors:
Function IXRouter01.WETH() (Strategy_Arthswap.sol#35) is not in mixedCase
Parameter Strategy.deposit(address,uint256)._wantAmt (Strategy_Arthswap.sol#1098) is not in mixedCase
Parameter Strategy.withdraw(address,uint256)._wantAmt (Strategy_Arthswap.sol#1153) is not in mixedCase
Parameter Strategy.buyBack(uint256)._earnedAmt (Strategy_Arthswap.sol#1291) is not in mixedCase
Parameter Strategy.distributeFees(uint256)._earnedAmt (Strategy_Arthswap.sol#1354) is not in mixedCase
Parameter Strategy.setEntranceFeeFactor(uint256)._entranceFeeFactor (Strategy_Arthswap.sol#1421) is not in mixedCase
Parameter Strategy.setControllerFee(uint256)._controllerFee (Strategy_Arthswap.sol#1427) is not in mixedCase
```

```
INFO:Detectors:
Redundant expression "this (Strategy_Arthswap.sol#894)" inContext (Strategy_Arthswap.sol#888-897)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IXRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Strategy_Arthswap.sol#
40) is too similar to IXRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Strategy_
Arthswap.sol#41)
Variable Strategy.earnedToToken0Path (Strategy_Arthswap.sol#1085) is too similar to Strategy.earnedToToken1Path (Strategy_Arthswap.sol#1
086)
Variable Strategy.token0Address (Strategy_Arthswap.sol#1044) is too similar to Strategy.token1Address (Strategy_Arthswap.sol#1045)
Variable Strategy.token0ToEarnedPath (Strategy_Arthswap.sol#1087) is too similar to Strategy.token1ToEarnedPath (Strategy_Arthswap.sol#1
088)
Variable Strategy_Arthswap.constructor(address[],address[],bool,bool,uint256,address[],address[],address[],address[],address[],address[]
,uint256,uint256,uint256)._earnedToToken0Path (Strategy_Arthswap.sol#1510) is too similar to Strategy_Arthswap.constructor(address[],add
ress[],bool,bool,uint256,address[],address[],address[],address[],address[],address[],uint256,uint256,uint256)._earnedToToken1Path (Strat
egy_Arthswap.sol#1511)
Variable Strategy_Arthswap.constructor(address[],address[],bool,bool,uint256,address[],address[],address[],address[],address[],address[]
,uint256,uint256,uint256)._token0ToEarnedPath (Strategy_Arthswap.sol#1512) is too similar to Strategy_Arthswap.constructor(address[],add
ress[],bool,bool,uint256,address[],address[],address[],address[],address[],address[],uint256,uint256,uint256)._token1ToEarnedPath (Strat
egy_Arthswap.sol#1513)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
Strategy_Arthswap.slitherConstructorConstantVariables() (Strategy_Arthswap.sol#1499-1552) uses literals with too many digits:
        - buyBackAddress = 0x000000000000000000000000000000000000dEaD (Strategy_Arthswap.sol#1067)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Strategy.routerDeadlineDuration (Strategy_Arthswap.sol#1049) should be constant
Strategy.slippageFactor (Strategy_Arthswap.sol#1081) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
```

```
INFO:Detectors:
owner() should be declared external:
        - Ownable.owner() (Strategy_Arthswap.sol#919-921)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Strategy_Arthswap.sol#938-941)
paused() should be declared external:
        - Pausable.paused() (Strategy_Arthswap.sol#980-982)
farm() should be declared external:
        - Strategy.farm() (Strategy_Arthswap.sol#1136-1138)
earn() should be declared external:
        - Strategy.earn() (Strategy_Arthswap.sol#1199-1289)
convertDustToEarned() should be declared external:
        - Strategy.convertDustToEarned() (Strategy_Arthswap.sol#1368-1411)
pause() should be declared external:
        - Strategy.pause() (Strategy_Arthswap.sol#1413-1415)
setEntranceFeeFactor(uint256) should be declared external:
        - Strategy.setEntranceFeeFactor(uint256) (Strategy_Arthswap.sol#1421-1425)
setControllerFee(uint256) should be declared external:
        - Strategy.setControllerFee(uint256) (Strategy_Arthswap.sol#1427-1430)
```

```
setGov(address) should be declared external:
        - Strategy.setGov(address) (Strategy_Arthswap.sol#1432-1434)
setDepositFeeFactor(uint256) should be declared external:
        - Strategy.setDepositFeeFactor(uint256) (Strategy_Arthswap.sol#1436-1440)
setWithdrawFeeFactor(uint256) should be declared external:
        - Strategy.setWithdrawFeeFactor(uint256) (Strategy_Arthswap.sol#1442-1446)
setbuyBackRate(uint256) should be declared external:
        - Strategy.setbuyBackRate(uint256) (Strategy_Arthswap.sol#1448-1451)
setBuybackRouterAddress(address) should be declared external:
        - Strategy.setBuybackRouterAddress(address) (Strategy_Arthswap.sol#1453-1455)
inCaseTokensGetStuck(address,uint256,address) should be declared external:
        - Strategy.inCaseTokensGetStuck(address,uint256,address) (Strategy_Arthswap.sol#1457-1465)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Strategy_Arthswap.sol analyzed (14 contracts with 75 detectors), 83 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> TimelockController.sol

```
INFO:Detectors:
Reentrancy in TimelockController._call(bytes32,uint256,address,uint256,bytes) (TimelockController.sol#1587-1599):
        External calls:
        - (success) = target.call{value: value}(data) (TimelockController.sol#1595)
        Event emitted after the call(s):
        - CallExecuted(id,index,target,value,data) (TimelockController.sol#1598)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
TimelockController.isOperationPending(bytes32) (TimelockController.sol#1345-1347) uses timestamp for comparisons
        Dangerous comparisons:
        - _timestamps[id] > _DONE_TIMESTAMP (TimelockController.sol#1346)
TimelockController.isOperationReady(bytes32) (TimelockController.sol#1352-1357) uses timestamp for comparisons
        Dangerous comparisons:
        - _timestamps[id] > _DONE_TIMESTAMP && _timestamps[id] <= block.timestamp (TimelockController.sol#1354-1356)
TimelockController.isOperationDone(bytes32) (TimelockController.sol#1362-1364) uses timestamp for comparisons
        Dangerous comparisons:
        - _timestamps[id] == _DONE_TIMESTAMP (TimelockController.sol#1363)
TimelockController._schedule(bytes32,uint256) (TimelockController.sol#1478-1486) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_timestamps[id] == 0,TimelockController: operation already scheduled) (TimelockController.sol#1479-1482)
TimelockController._beforeCall(bytes32) (TimelockController.sol#1564-1569) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(predecessor == bytes32(0) || isOperationDone(predecessor),TimelockController: missing dependency) (Timelo
ckController.sol#1565-1568)
TimelockController.scheduleSet(address,uint256,uint256,bool,bytes32,bytes32) (TimelockController.sol#1633-1668) uses timestamp for compa
risons
        Dangerous comparisons:
        - require(bool,string)(_timestamps[id] == 0,TimelockController: operation already scheduled) (TimelockController.sol#1653-1656)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (TimelockController.sol#179-190) uses assembly
        - INLINE ASM (TimelockController.sol#186-188)
Address._verifyCallResult(bool,bytes,string) (TimelockController.sol#383-404) uses assembly
        - INLINE ASM (TimelockController.sol#396-399)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

```
INFO:Detectors:
Address.functionCall(address,bytes) (TimelockController.sol#240-245) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (TimelockController.sol#272-284) is never used and should be removed
Address.functionDelegateCall(address,bytes) (TimelockController.sol#353-363) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (TimelockController.sol#371-381) is never used and should be removed
Address.functionStaticCall(address,bytes) (TimelockController.sol#316-327) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (TimelockController.sol#335-345) is never used and should be removed
```

```
SafeMath.sub(uint256,uint256,string) (TimelockController.sol#48-57) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (TimelockController.sol#208-220):
        - (success) = recipient.call{value: amount}() (TimelockController.sol#215)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (TimelockController.sol#292-308):
        - (success,returndata) = target.call{value: value}(data) (TimelockController.sol#305-306)
Low level call in Address.functionStaticCall(address,bytes,string) (TimelockController.sol#335-345):
        - (success,returndata) = target.staticcall(data) (TimelockController.sol#343)
Low level call in Address.functionDelegateCall(address,bytes,string) (TimelockController.sol#371-381):
        - (success,returndata) = target.delegatecall(data) (TimelockController.sol#379)
Low level call in TimelockController._call(bytes32,uint256,address,uint256,bytes) (TimelockController.sol#1587-1599):
        - (success) = target.call{value: value}(data) (TimelockController.sol#1595)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter TimelockController.setDevWalletAddress(address)._devWalletAddress (TimelockController.sol#1623) is not in mixedCase
Parameter TimelockController.scheduleSet(address,uint256,uint256,bool,bytes32,bytes32)._nativefarmAddress (TimelockController.sol#1634)
is not in mixedCase
Parameter TimelockController.scheduleSet(address,uint256,uint256,bool,bytes32,bytes32)._pid (TimelockController.sol#1635) is not in mixe
dCase
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (TimelockController.sol#1016)" inContext (TimelockController.sol#1010-1019)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
getRoleMemberCount(bytes32) should be declared external:
        - AccessControl.getRoleMemberCount(bytes32) (TimelockController.sol#1084-1086)
getRoleMember(bytes32,uint256) should be declared external:
        - AccessControl.getRoleMember(bytes32,uint256) (TimelockController.sol#1100-1106)
getRoleAdmin(bytes32) should be declared external:
        - AccessControl.getRoleAdmin(bytes32) (TimelockController.sol#1114-1116)
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (TimelockController.sol#1128-1135)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (TimelockController.sol#1146-1153)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (TimelockController.sol#1169-1176)
getTimestamp(bytes32) should be declared external:
        - TimelockController.getTimestamp(bytes32) (TimelockController.sol#1370-1372)
getMinDelay() should be declared external:
        - TimelockController.getMinDelay() (TimelockController.sol#1379-1381)
schedule(address,uint256,bytes,bytes32,bytes32,uint256) should be declared external:
        - TimelockController.schedule(address,uint256,bytes,bytes32,bytes32,uint256) (TimelockController.sol#1420-1431)
scheduleBatch(address[],uint256[],bytes[],bytes32,bytes32,uint256) should be declared external:
        - TimelockController.scheduleBatch(address[],uint256[],bytes[],bytes32,bytes32,uint256) (TimelockController.sol#1442-1473)
cancel(bytes32) should be declared external:
        - TimelockController.cancel(bytes32) (TimelockController.sol#1495-1503)
```

```
execute(address,uint256,bytes,bytes32,bytes32) should be declared external:
        - TimelockController.execute(address,uint256,bytes,bytes32,bytes32) (TimelockController.sol#1514-1525)
executeBatch(address[],uint256[],bytes[],bytes32,bytes32) should be declared external:
        - TimelockController.executeBatch(address[],uint256[],bytes[],bytes32,bytes32) (TimelockController.sol#1536-1559)
setDevWalletAddress(address) should be declared external:
        - TimelockController.setDevWalletAddress(address) (TimelockController.sol#1623-1628)
scheduleSet(address,uint256,uint256,bool,bytes32,bytes32) should be declared external:
        - TimelockController.scheduleSet(address,uint256,uint256,bool,bytes32,bytes32) (TimelockController.sol#1633-1668)
executeSet(address,uint256,uint256,bool,bytes32,bytes32) should be declared external:
        - TimelockController.executeSet(address,uint256,uint256,bool,bytes32,bytes32) (TimelockController.sol#1670-1693)
withdrawFTM() should be declared external:
        - TimelockController.withdrawFTM() (TimelockController.sol#1698-1701)
withdrawBEP20(address) should be declared external:
        - TimelockController.withdrawBEP20(address) (TimelockController.sol#1703-1708)
add(address,address,bool,address) should be declared external:
        - TimelockController.add(address,address,bool,address) (TimelockController.sol#1710-1717)
earn(address) should be declared external:
        - TimelockController.earn(address) (TimelockController.sol#1719-1721)
farm(address) should be declared external:
        - TimelockController.farm(address) (TimelockController.sol#1723-1725)
pause(address) should be declared external:
        - TimelockController.pause(address) (TimelockController.sol#1727-1729)
unpause(address) should be declared external:
        - TimelockController.unpause(address) (TimelockController.sol#1731-1733)
rebalance(address,uint256,uint256) should be declared external:
        - TimelockController.rebalance(address,uint256,uint256) (TimelockController.sol#1735-1741)
deleverageOnce(address) should be declared external:
        - TimelockController.deleverageOnce(address) (TimelockController.sol#1744-1746)
wrapFTM(address) should be declared external:
        - TimelockController.wrapFTM(address) (TimelockController.sol#1748-1750)
noTimeLockFunc1(address) should be declared external:
        - TimelockController.noTimeLockFunc1(address) (TimelockController.sol#1753-1758)
noTimeLockFunc2(address) should be declared external:
```

```
        - TimelockController.noTimeLockFunc1(address) (TimelockController.sol#1753-1758)
noTimeLockFunc2(address) should be declared external:
        - TimelockController.noTimeLockFunc2(address) (TimelockController.sol#1760-1765)
noTimeLockFunc3(address) should be declared external:
        - TimelockController.noTimeLockFunc3(address) (TimelockController.sol#1767-1772)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:TimelockController.sol analyzed (11 contracts with 75 detectors), 106 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> xALNR.sol

```
INFO:Detectors:
ERC20.constructor(string,string).name (xALNR.sol#246) shadows:
        - ERC20.name() (xALNR.sol#255-257) (function)
ERC20.constructor(string,string).symbol (xALNR.sol#246) shadows:
        - ERC20.symbol() (xALNR.sol#263-265) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
AlnrStakingToken.setDelayToWithdraw(uint256) (xALNR.sol#674-677) should emit an event for:
        - delayToWithdraw = second (xALNR.sol#676)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
AlnrStakingToken.setAdmin(address)._admin (xALNR.sol#995) lacks a zero-check on :
                - admin = _admin (xALNR.sol#997)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in AlnrStakingToken.enter(uint256) (xALNR.sol#680-698):
        External calls:
        - alnr.transferFrom(msg.sender,address(this),_amount) (xALNR.sol#695)
        State variables written after the call(s):
        - _stakedTime[msg.sender] = block.timestamp (xALNR.sol#697)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
AlnrStakingToken.canWithdraw(address) (xALNR.sol#664-672) uses timestamp for comparisons
        Dangerous comparisons:
        - _stakedTime[account] == 0 (xALNR.sol#665)
        - _stakedTime[account] + delayToWithdraw < block.timestamp (xALNR.sol#668)
AlnrStakingToken.delegateBySig(address,uint256,uint256,uint8,bytes32,bytes32) (xALNR.sol#838-870) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp <= expiry,ALNR::delegateBySig: signature expired) (xALNR.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (xALNR.sol#34-43) uses assembly
        - INLINE ASM (xALNR.sol#41)
Address._functionCallWithValue(address,bytes,uint256,string) (xALNR.sol#127-148) uses assembly
        - INLINE ASM (xALNR.sol#140-143)
AlnrStakingToken.getChainId() (xALNR.sol#988-992) uses assembly
```

```
AlnrStakingToken.getChainId() (xALNR.sol#988-992) uses assembly
        - INLINE ASM (xALNR.sol#990)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (xALNR.sol#127-148) is never used and should be removed
Address.functionCall(address,bytes) (xALNR.sol#87-89) is never used and should be removed
Address.functionCall(address,bytes,string) (xALNR.sol#97-99) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (xALNR.sol#112-114) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (xALNR.sol#122-125) is never used and should be removed
Address.isContract(address) (xALNR.sol#34-43) is never used and should be removed
Address.sendValue(address,uint256) (xALNR.sol#61-67) is never used and should be removed
Context._msgData() (xALNR.sol#10-13) is never used and should be removed
ERC20._setupDecimals(uint8) (xALNR.sol#477-479) is never used and should be removed
SafeMath.mod(uint256,uint256) (xALNR.sol#619-621) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (xALNR.sol#635-638) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (xALNR.sol#61-67):
        - (success) = recipient.call{value: amount}() (xALNR.sol#65)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (xALNR.sol#127-148):
        - (success,returndata) = target.call{value: weiValue}(data) (xALNR.sol#131)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter AlnrStakingToken.enter(uint256)._amount (xALNR.sol#680) is not in mixedCase
Parameter AlnrStakingToken.leave(uint256)._share (xALNR.sol#701) is not in mixedCase
Function AlnrStakingToken.ALNRBalance(address) (xALNR.sol#713-717) is not in mixedCase
Parameter AlnrStakingToken.ALNRBalance(address)._account (xALNR.sol#713) is not in mixedCase
Parameter AlnrStakingToken.xALNRForALNR(uint256)._xALNRAmount (xALNR.sol#720) is not in mixedCase
Function AlnrStakingToken.ALNRForxALNR(uint256) (xALNR.sol#726-735) is not in mixedCase
```

```
Parameter AlnrStakingToken.ALNRForxALNR(uint256)._alnrAmount (xALNR.sol#726) is not in mixedCase
Parameter AlnrStakingToken.burn(address,uint256)._from (xALNR.sol#766) is not in mixedCase
Parameter AlnrStakingToken.burn(address,uint256)._amount (xALNR.sol#766) is not in mixedCase
Parameter AlnrStakingToken.mint(address,uint256)._amount (xALNR.sol#771) is not in mixedCase
Parameter AlnrStakingToken.setAdmin(address)._admin (xALNR.sol#995) is not in mixedCase
Variable AlnrStakingToken._stakedTime (xALNR.sol#651) is not in mixedCase
Variable AlnrStakingToken._delegates (xALNR.sol#737) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (xALNR.sol#11)" inContext (xALNR.sol#5-14)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable AlnrStakingToken.ALNRForxALNR(uint256)._alnrAmount (xALNR.sol#726) is too similar to AlnrStakingToken.ALNRBalance(address).xALN
RAmount (xALNR.sol#714)
Variable AlnrStakingToken.xALNRForALNR(uint256)._xALNRAmount (xALNR.sol#720) is too similar to AlnrStakingToken.ALNRForxALNR(uint256).xA
LNRAmount_ (xALNR.sol#726)
Variable AlnrStakingToken.ALNRForxALNR(uint256)._alnrAmount (xALNR.sol#726) is too similar to AlnrStakingToken.ALNRBalance(address).alnr
Amount_ (xALNR.sol#713)
Variable AlnrStakingToken.ALNRForxALNR(uint256)._alnrAmount (xALNR.sol#726) is too similar to AlnrStakingToken.xALNRForALNR(uint256).aln
rAmount_ (xALNR.sol#720)
Variable AlnrStakingToken.ALNRBalance(address).alnrAmount_ (xALNR.sol#713) is too similar to AlnrStakingToken.ALNRBalance(address).xALNR
Amount (xALNR.sol#714)
Variable AlnrStakingToken.xALNRForALNR(uint256).alnrAmount_ (xALNR.sol#720) is too similar to AlnrStakingToken.ALNRBalance(address).xALN
RAmount (xALNR.sol#714)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
symbol() should be declared external:
        - ERC20.symbol() (xALNR.sol#263-265)
decimals() should be declared external:
        - ERC20.decimals() (xALNR.sol#280-282)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (xALNR.sol#314-316)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (xALNR.sol#325-328)
```

```
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (xALNR.sol#360-363)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (xALNR.sol#379-382)
stakedTime(address) should be declared external:
        - AlnrStakingToken.stakedTime(address) (xALNR.sol#660-662)
enter(uint256) should be declared external:
        - AlnrStakingToken.enter(uint256) (xALNR.sol#680-698)
leave(uint256) should be declared external:
        - AlnrStakingToken.leave(uint256) (xALNR.sol#701-710)
setAdmin(address) should be declared external:
        - AlnrStakingToken.setAdmin(address) (xALNR.sol#995-998)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:xALNR.sol analyzed (6 contracts with 75 detectors), 59 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Nika.sol

```
INFO:Detectors:
Nika.allowance(address,address).owner (Nika.sol#1244) shadows:
        - Ownable.owner() (Nika.sol#30-32) (function)
Nika._approve(address,address,uint256).owner (Nika.sol#1436) shadows:
        - Ownable.owner() (Nika.sol#30-32) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Nika.constructor(address,address,address[])._marketingWallet (Nika.sol#1176) lacks a zero-check on :
                - marketingWallet = _marketingWallet (Nika.sol#1180)
Nika.constructor(address,address,address[])._liquidityWallet (Nika.sol#1177) lacks a zero-check on :
                - liquidityWallet = _liquidityWallet (Nika.sol#1181)
Nika.setWallet(address,address)._marketingWallet (Nika.sol#1588) lacks a zero-check on :
                - marketingWallet = _marketingWallet (Nika.sol#1591)
Nika.setWallet(address,address)._liquidityWallet (Nika.sol#1589) lacks a zero-check on :
                - liquidityWallet = _liquidityWallet (Nika.sol#1592)
Nika.updateUniswapV2Router(address)._uniswapV2Pair (Nika.sol#1633-1634) lacks a zero-check on :
                - uniswapV2Pair = _uniswapV2Pair (Nika.sol#1635)
DividendTracker.constructor(address,address)._tokenAddress (Nika.sol#1820) lacks a zero-check on :
                - tokenAddress = _tokenAddress (Nika.sol#1822)
DividendTracker.constructor(address,address)._uniswapRouter (Nika.sol#1820) lacks a zero-check on :
                - UNISWAPROUTER = _uniswapRouter (Nika.sol#1823)
DividendTracker.manualSendDividend(uint256,address).holder (Nika.sol#1885) lacks a zero-check on :
                - address(holder).transfer(amount) (Nika.sol#1890)
                - address(holder).transfer(contractETHBalance) (Nika.sol#1890)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in Nika._executeSwap(uint256,uint256) (Nika.sol#1494-1550):
        External calls:
        - swapTokensForNative(swapTokensTotal) (Nika.sol#1519)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokens,0,path,address(this),block.timestamp) (Nika.
l#1467-1473)
```

```
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (Nika.sol#1443)
                - _approve(sender,_msgSender(),currentAllowance - amount) (Nika.sol#1310)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Address.isContract(address) (Nika.sol#505-515) uses assembly
        - INLINE ASM (Nika.sol#511-513)
Address.verifyCallResult(bool,bytes,string) (Nika.sol#674-694) uses assembly
        - INLINE ASM (Nika.sol#686-689)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address.functionCall(address,bytes) (Nika.sol#558-560) is never used and should be removed
Address.functionCall(address,bytes,string) (Nika.sol#568-574) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (Nika.sol#587-593) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (Nika.sol#601-612) is never used and should be removed
Address.functionDelegateCall(address,bytes) (Nika.sol#647-649) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (Nika.sol#657-666) is never used and should be removed
Address.functionStaticCall(address,bytes) (Nika.sol#620-622) is never used and should be removed
```

```
SafeMath.tryAdd(uint256,uint256) (Nika.sol#702-708) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Nika.sol#744-749) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (Nika.sol#756-761) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Nika.sol#727-737) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Nika.sol#715-720) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version=0.8.0 (Nika.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) (Nika.sol#533-538):
        - (success) = recipient.call{value: amount}() (Nika.sol#536)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (Nika.sol#601-612):
        - (success,returndata) = target.call{value: value}(data) (Nika.sol#610)
Low level call in Address.functionStaticCall(address,bytes,string) (Nika.sol#630-639):
        - (success,returndata) = target.staticcall(data) (Nika.sol#637)
Low level call in Address.functionDelegateCall(address,bytes,string) (Nika.sol#657-666):
        - (success,returndata) = target.delegatecall(data) (Nika.sol#664)
Low level call in Nika._executeSwap(uint256,uint256) (Nika.sol#1494-1550):
        - (success) = address(dividendTracker).call{value: nativeDividends}() (Nika.sol#1543-1545)
Low level call in DividendTracker._withdrawDividendOfUser(address) (Nika.sol#1949-1970):
        - (success) = account.call{gas: 3000,value: _withdrawableDividend}() (Nika.sol#1958-1961)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

```
INFO:Detectors:
Nika (Nika.sol#1111-1781) should inherit from IERC20Metadata (Nika.sol#148-163)
DividendTracker (Nika.sol#1783-2133) should inherit from IERC20Metadata (Nika.sol#148-163)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-inheritance
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (Nika.sol#972) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (Nika.sol#974) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (Nika.sol#1005) is not in mixedCase
```

```
INFO:Detectors:
Variable IUniswapV2Router02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Nika.sol#1057)
s too similar to IUniswapV2Router02.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Nika.s
#1058)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
Nika.constructor(address,address,address[]) (Nika.sol#1175-1214) uses literals with too many digits:
        - _mint(owner(),1000000 * (10 ** 18)) (Nika.sol#1213)
Nika.slitherConstructorVariables() (Nika.sol#1111-1781) uses literals with too many digits:
        - DEAD = 0x000000000000000000000000000000000000dEaD (Nika.sol#1113)
Nika.slitherConstructorVariables() (Nika.sol#1111-1781) uses literals with too many digits:
        - ZERO = 0x0000000000000000000000000000000000000000 (Nika.sol#1114)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Nika.ZERO (Nika.sol#1114) is never used in Nika (Nika.sol#1111-1781)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
DividendTracker._name (Nika.sol#1786) should be constant
DividendTracker._symbol (Nika.sol#1787) should be constant
DividendTracker.lastProcessedIndex (Nika.sol#1789) should be constant
Nika.DEAD (Nika.sol#1113) should be constant
Nika.UNISWAPROUTER (Nika.sol#1112) should be constant
Nika.ZERO (Nika.sol#1114) should be constant
Nika._name (Nika.sol#1116) should be constant
Nika._symbol (Nika.sol#1117) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Nika.sol#49-51)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Nika.sol#57-60)
name() should be declared external:
        - ERC20.name() (Nika.sol#191-193)
symbol() should be declared external:
        - ERC20.symbol() (Nika.sol#199-201)
```

```
        - Nika.getAccountInfo(address) (Nika.sol#1671-1683)
getLastClaimTime(address) should be declared external:
        - Nika.getLastClaimTime(address) (Nika.sol#1685-1687)
isExcludedFromMaxTx(address) should be declared external:
        - Nika.isExcludedFromMaxTx(address) (Nika.sol#1723-1725)
isExcludedFromMaxWallet(address) should be declared external:
        - Nika.isExcludedFromMaxWallet(address) (Nika.sol#1742-1748)
blackList(address) should be declared external:
        - Nika.blackList(address) (Nika.sol#1758-1762)
removeFromBlacklist(address) should be declared external:
        - Nika.removeFromBlacklist(address) (Nika.sol#1764-1768)
blackListMany(address[]) should be declared external:
        - Nika.blackListMany(address[]) (Nika.sol#1770-1774)
unBlackListMany(address[]) should be declared external:
        - Nika.unBlackListMany(address[]) (Nika.sol#1776-1780)
isExcludedFromDividends(address) should be declared external:
        - DividendTracker.isExcludedFromDividends(address) (Nika.sol#1877-1883)
processAccount(address) should be declared external:
        - DividendTracker.processAccount(address) (Nika.sol#1935-1947)
compoundAccount(address) should be declared external:
        - DividendTracker.compoundAccount(address) (Nika.sol#1972-1984)
withdrawnDividendOf(address) should be declared external:
        - DividendTracker.withdrawnDividendOf(address) (Nika.sol#2041-2047)
getAccountInfo(address) should be declared external:
        - DividendTracker.getAccountInfo(address) (Nika.sol#2059-2082)
getLastClaimTime(address) should be declared external:
        - DividendTracker.getLastClaimTime(address) (Nika.sol#2084-2086)
name() should be declared external:
        - DividendTracker.name() (Nika.sol#2088-2090)
symbol() should be declared external:
        - DividendTracker.symbol() (Nika.sol#2092-2094)
decimals() should be declared external:
        - DividendTracker.decimals() (Nika.sol#2096-2098)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Nika.sol analyzed (12 contracts with 75 detectors), 173 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
root@server:/chetan/gaza/mycontracts#
```

Solidity Static Analysis

NativeFarm.sol

Security

Check-effects-interaction:

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

more

Pos: 725:4:

Check-effects-interaction:

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

more

Pos: 1609:4:

Block timestamp:

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

more

Pos: 1489:33:

Low level calls:

Use of "delegatecall": should be avoided whenever possible. External code, that is called can change the state of the calling contract and send ether from the caller's balance. If this is wanted behaviour, use the Solidity library feature if possible.

more

Pos: 812:50:

Gas & Economy

Gas costs:

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

Pos: 286:4:

Gas costs:

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

Pos: 1609:4:

Gas costs:

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

Pos: 1640:4:

Email: audit@EtherAuthority.io

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 44:4:

Constant/View/Pure functions:

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

more

Pos: 1750:4:

Similar variable names:

NativeFarm.setwithdrawalFeeFactor(uint256) : Variables have very similar names "withdrawalFeeFactor" and "withdrawalFeeFactorLL". Note: Modifiers are currently not considered by this static analysis.

Pos: 1766:36:

Guard conditions:

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

more

Pos: 1771:8:

NATIVEToken.sol

Gas & Economy

Gas costs:

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

Pos: 286:4:

Gas costs:

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

Pos: 658:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 44:4:

Similar variable names:

ALNR.removeMinter(address) : Variables have very similar names "minters" and "_minter". Note: Modifiers are currently not considered by this static analysis.
Pos: 668:16:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 659:8:

Strategy.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 524:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Strategy._farm(): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1141:4:

Block timestamp:

Use of "now": "now" does not mean current time. "now" is an alias for "block.timestamp". "block.timestamp" can be influenced by miners to a certain degree, be careful.
more
Pos: 1226:20:

Low level calls:

Use of "delegatecall": should be avoided whenever possible. External code, that is called can change the state of the calling contract and send ether from the caller's balance. If this is wanted behaviour, use the Solidity library feature if possible.
more
Pos: 611:50:

Miscellaneous

Constant/View/Pure functions:

SafeMath.sub(uint256,uint256) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 266:4:

Constant/View/Pure functions:

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

more

Pos: 1355:4:

Constant/View/Pure functions:

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

more

Pos: 1458:4:

Similar variable names:

Strategy.deposit(address,uint256) : Variables have very similar names "depositFeeFactor" and "depositFeeFactorLL". Note: Modifiers are currently not considered by this static analysis.

Pos: 1115:36:

Similar variable names:

Strategy.earn() : Variables have very similar names "token0Amt" and "token1Amt". Note: Modifiers are currently not considered by this static analysis.

Pos: 1266:12:

Guard conditions:

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

more

Pos: 1423:8:

Guard conditions:

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

more

Pos: 1424:8:

Strategy_Arthswap.sol

Security

Check-effects-interaction:

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

more

Pos: 523:4:

Check-effects-interaction:

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

more

Pos: 1368:4:

Block timestamp:

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

more

Pos: 1408:16:

Low level calls:

Use of "delegatecall": should be avoided whenever possible. External code, that is called can change the state of the calling contract and send ether from the caller's balance. If this is wanted behaviour, use the Solidity library feature if possible.

more

Pos: 610:50:

Gas & Economy

Gas costs:

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

Pos: 947:4:

Gas costs:

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

Pos: 1457:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 265:4:

Constant/View/Pure functions:

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

more

Pos: 1457:4:

Similar variable names:

Strategy_Arthswap.
(address[],address[],bool,bool,uint256,address[],address[],address[],address[],address[],address[],uint256,uint256,uint256)
: Variables have very similar names "token0Address" and "token1Address". Note: Modifiers are currently not considered by this static analysis.

Pos: 1531:8:

Guard conditions:

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

more

Pos: 1463:8:

Data truncated:

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

Pos: 309:16:

TimelockController.sol

Security

Transaction origin:

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

more

Pos: 1625:16:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in TimelockController.executeSet(address,uint256,uint256,bool,bytes32,bytes32): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 1670:4:

Block timestamp:

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

more

Pos: 1356:31:

Low level calls:

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

more

Pos: 1595:27:

Gas & Economy

Gas costs:

Gas requirement of function TimelockController.hasRole is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1076:4:

Gas costs:

Gas requirement of function TimelockController.scheduleSet is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1633:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 1555:8:

Miscellaneous

Constant/View/Pure functions:

SafeMath.sub(uint256,uint256) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 34:4:

Constant/View/Pure functions:

TimelockController.hashOperationBatch(address[],uint256[],bytes[],bytes32,bytes32) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1401:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 1704:8:

Delete from dynamic array:

Using "delete" on an array leaves a gap. The length of the array remains the same. If you want to remove the empty position you need to shift items manually and update the "length" property.
more
Pos: 1500:8:

xALNR.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address._functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability.
more
Pos: 127:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in AlnrStakingToken.enter(uint256): Could potentially lead to re-entrancy vulnerability.
more
Pos: 680:4:

Inline assembly:

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.
more
Pos: 990:8:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 668:56:

Gas & Economy

Gas costs:

Gas requirement of function AlnrStakingToken.name is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 255:4:

Gas costs:

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

Gas costs:

Gas requirement of function AlnrStakingToken.xALNRForALNR is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 720:4:

Miscellaneous

Constant/View/Pure functions:

Address.isContract(address) : Is constant but potentially should not be.
more
Pos: 34:4:

Constant/View/Pure functions:

AlnrStakingToken.ALNRForxALNR(uint256) : Is constant but potentially should not be.
more
Pos: 726:4:

Constant/View/Pure functions:

AlnrStakingToken.getChainId() : Is constant but potentially should not be.
more
Pos: 988:4:

Similar variable names:

AlnrStakingToken._writeCheckpoint(address,uint32,uint256,uint256) : Variables have very similar names "checkpoints" and "nCheckpoints".
Pos: 976:35:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 984:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 996:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 918:36:

Nika.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 601:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Nika.(address,address,address[]): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1175:4:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 1385:27:

Low level calls:

Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.
more
Pos: 1958:31:

Gas & Economy

Gas costs:

Gas requirement of function Nika.setAutomatedMarketMakerPair is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1595:4:

Gas costs:

Gas requirement of function Nika.setFee is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 1603:4:

Gas costs:

Gas requirement of function Nika.transferFrom is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 2125:4:

For loop over dynamic array:

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

more

Pos: 1777:8:

ERC

ERC20:

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

more

Pos: 951:4:

Miscellaneous

Constant/View/Pure functions:

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

more

Pos: 2059:4:

Similar variable names:

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

Pos: 1924:30:

Similar variable names:

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

Pos: 1927:18:

No return:

DividendTracker.approve(address,uint256): Defines a return type but never explicitly returns a value.

Pos: 2121:4:

No return:

DividendTracker.transferFrom(address,address,uint256): Defines a return type but never explicitly returns a value.

Pos: 2125:4:

Guard conditions:

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

more

Pos: 1359:8:

Solhint Linter

NativeFarm.sol

```
NativeFarm.sol:3:1: Error: Compiler version 0.6.12 does not satisfy
the r semver requirement
NativeFarm.sol:591:24: Error: Code contains empty blocks
NativeFarm.sol:739:13: Error: Avoid using low level calls.
NativeFarm.sol:1443:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1445:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1447:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1449:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1453:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1454:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1456:29: Error: Constant name must be in capitalized
SNAKE_CASE
NativeFarm.sol:1489:34: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1489:64: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1536:13: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1537:69: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1538:13: Error: Variable name must be in mixedCase
NativeFarm.sol:1578:13: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1583:35: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1586:65: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1590:9: Error: Variable name must be in mixedCase
NativeFarm.sol:1605:31: Error: Avoid making time-based decisions in
your business logic
NativeFarm.sol:1729:46: Error: Variable name must be in mixedCase
NativeFarm.sol:1730:9: Error: Variable name must be in mixedCase
NativeFarm.sol:1740:33: Error: Variable name must be in mixedCase
```

NATIVEToken.sol

```
NATIVEToken.sol:3:1: Error: Compiler version ^0.6.12 does not satisfy
the r semver requirement
NATIVEToken.sol:588:24: Error: Code contains empty blocks
```

Strategy.sol

```
Strategy.sol:3:1: Error: Compiler version ^0.6.12 does not satisfy
the r semver requirement
Strategy.sol:36:5: Error: Function name must be in mixedCase
Strategy.sol:538:13: Error: Avoid using low level calls.
Strategy.sol:1033:1: Error: Contract has 31 states declarations but
allowed no more than 15
Strategy.sol:1054:20: Error: Variable name must be in mixedCase
Strategy.sol:1062:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1063:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1066:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1067:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1068:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1071:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1072:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1075:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1076:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1079:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1080:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1083:29: Error: Constant name must be in capitalized
SNAKE_CASE
Strategy.sol:1091:22: Error: Variable name must be in mixedCase
Strategy.sol:1099:22: Error: Variable "_userAddress" is unused
Strategy.sol:1154:23: Error: Variable "_userAddress" is unused
Strategy.sol:1226:21: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1229:28: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1247:17: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1259:17: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1283:17: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1287:24: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1315:21: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1333:21: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1348:17: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1390:17: Error: Avoid making time-based decisions in
your business logic
Strategy.sol:1409:17: Error: Avoid making time-based decisions in
your business logic
```

Strategy_Arthswap.sol

```
Strategy_Arthswap.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
Strategy_Arthswap.sol:35:5: Error: Function name must be in mixedCase
Strategy_Arthswap.sol:537:13: Error: Avoid using low level calls.
Strategy_Arthswap.sol:1032:1: Error: Contract has 31 states
declarations but allowed no more than 15
Strategy_Arthswap.sol:1053:20: Error: Variable name must be in
mixedCase
Strategy_Arthswap.sol:1061:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1062:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1065:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1066:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1067:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1070:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1071:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1074:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1075:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1078:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1079:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1082:29: Error: Constant name must be in
capitalized SNAKE_CASE
Strategy_Arthswap.sol:1090:22: Error: Variable name must be in
mixedCase
Strategy_Arthswap.sol:1098:22: Error: Variable "_userAddress" is
unused
Strategy_Arthswap.sol:1153:23: Error: Variable "_userAddress" is
unused
Strategy_Arthswap.sol:1225:21: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1228:28: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1246:17: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1258:17: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1282:17: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1286:24: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1314:21: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1332:21: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1347:17: Error: Avoid making time-based
```

```
decisions in your business logic
Strategy_Arthswap.sol:1389:17: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1408:17: Error: Avoid making time-based
decisions in your business logic
Strategy_Arthswap.sol:1499:1: Error: Contract name must be in
CamelCase
```

TimelockController.sol

```
TimelockController.sol:3:1: Error: Compiler version 0.6.12 does not
satisfy the r semver requirement
TimelockController.sol:306:13: Error: Avoid using low level calls.
TimelockController.sol:1340:32: Error: Code contains empty blocks
TimelockController.sol:1356:32: Error: Avoid making time-based
decisions in your business logic
TimelockController.sol:1625:17: Error: Avoid to use tx.origin
TimelockController.sol:1658:40: Error: Avoid making time-based
decisions in your business logic
```

xALNR.sol

```
xALNR.sol:3:1: Error: Compiler version 0.6.12 does not satisfy the r
semver requirement
xALNR.sol:495:94: Error: Code contains empty blocks
xALNR.sol:668:57: Error: Avoid making time-based decisions in your
business logic
xALNR.sol:697:35: Error: Avoid making time-based decisions in your
business logic
xALNR.sol:702:42: Error: Use double quotes for string literals
xALNR.sol:713:5: Error: Function name must be in mixedCase
xALNR.sol:726:5: Error: Function name must be in mixedCase
xALNR.sol:868:17: Error: Avoid making time-based decisions in your
business logic
xALNR.sol:990:9: Error: Avoid using inline assembly. It is acceptable
only in rare cases
```

Nika.sol

```
Nika.sol:288:18: Error: Parse error: missing ';' at '{'
Nika.sol:329:18: Error: Parse error: missing ';' at '{'
Nika.sol:362:18: Error: Parse error: missing ';' at '{'
Nika.sol:411:18: Error: Parse error: missing ';' at '{'
Nika.sol:703:18: Error: Parse error: missing ';' at '{'
Nika.sol:716:18: Error: Parse error: missing ';' at '{'
Nika.sol:728:18: Error: Parse error: missing ';' at '{'
Nika.sol:745:18: Error: Parse error: missing ';' at '{'
Nika.sol:757:18: Error: Parse error: missing ';' at '{'
Nika.sol:853:18: Error: Parse error: missing ';' at '{'
Nika.sol:876:18: Error: Parse error: missing ';' at '{'
Nika.sol:902:18: Error: Parse error: missing ';' at '{'
```

Software analysis result:

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

Email: audit@EtherAuthority.io

Email: audit@EtherAuthority.io