AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Alpha Finance

DATED : 19 MAR 23'

AUDIT SUMMARY

Project name – Alpha Finance

Date: 19 March, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	3
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
a line by line code review has been performed by audit ace team.

2- BSC Testnet network:
all tests were done on Bsc Testnet network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

https://testnet.bscscan.com/token/0x4af3bc79fcd801278fbace36f59955010c102697

Token Information

Token Name : Alpha Finance

Token Symbol: ALPHA

Decimals: 18

Token Supply: 1,000,000,000

Token Address:
0x1907351Db830191ac40f0BCC49d84b1CA344c813

Checksum:
3ca541da30aa838ddec4f5958c6376076fc86407

Owner:
0x7261Ae7d5239474D965B0531f960034DCfFcb9E1
(at time of writing the audit)

TOKEN OVERVIEW

Fees:

Buy Fees: 0%

Sell Fees: 1%

Transfer Fees: 0%

Fees Privilige: none

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: none

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] Using throw

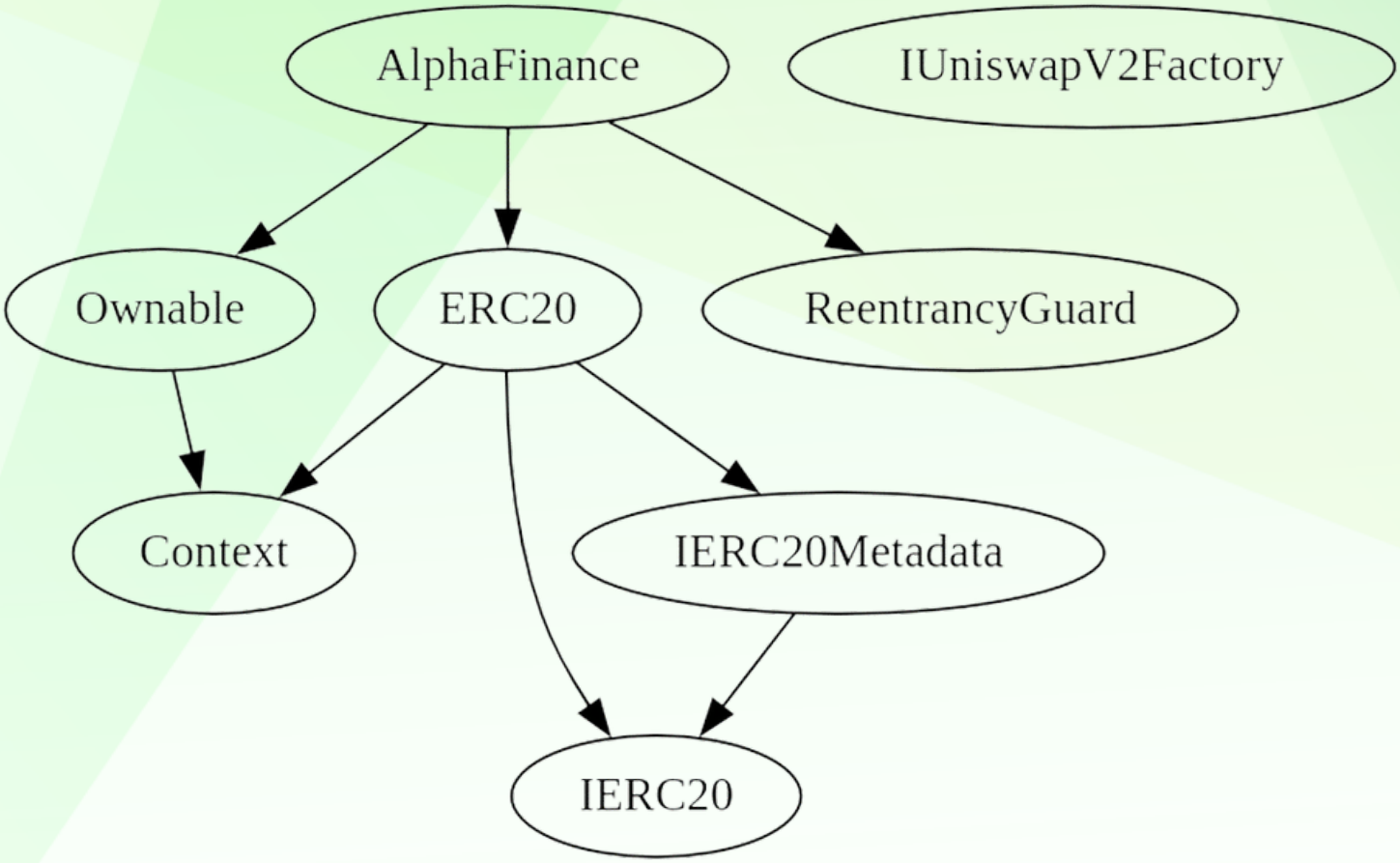
CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	0
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	3

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to change buy/sell/transfer fees (buy 0%, sell 1%, 0% transfer)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner is not able to mint new tokens

TOKEN DISTRIBUTION

It should be noted that the owner currently holds 100% of the total supply. However, information about the distribution of these tokens is not available, and it is recommended that investors exercise caution when considering this aspect.

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IUniswapV2Factory	Interface			
└	feeTo	External ❗		NO❗
└	feeToSetter	External ❗		NO❗
└	getPair	External ❗		NO❗
└	allPairs	External ❗		NO❗
└	allPairsLength	External ❗		NO❗
└	createPair	External ❗	🛑	NO❗
└	setFeeTo	External ❗	🛑	NO❗
└	setFeeToSetter	External ❗	🛑	NO❗
IUniswapV2Pair	Interface			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🛑	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗

CONTRACT ASSESMENT

└	swap	External ❗	🛑	NO❗
└	skim	External ❗	🛑	NO❗
└	sync	External ❗	🛑	NO❗
└	initialize	External ❗	🛑	NO❗
IUniswapV2Router01	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🛑	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🛑	NO❗
└	removeLiquidityETH	External ❗	🛑	NO❗
└	removeLiquidityWithPermit	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO❗
└	swapExactTokensForTokens	External ❗	🛑	NO❗
└	swapTokensForExactTokens	External ❗	🛑	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🛑	NO❗
└	swapExactTokensForETH	External ❗	🛑	NO❗
└	swapETHForExactTokens	External ❗	💵	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
IERC20Permit	Interface			
└	permit	External ❗	🛑	NO❗
└	nonces	External ❗		NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	

CONTRACT ASSESMENT

└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		
└	_revert	Private 🔐		
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeERC20	Library			
└	safeTransfer	Internal 🔒	🛑	
└	safeTransferFrom	Internal 🔒	🛑	
└	safeApprove	Internal 🔒	🛑	
└	safeIncreaseAllowance	Internal 🔒	🛑	
└	safeDecreaseAllowance	Internal 🔒	🛑	
└	safePermit	Internal 🔒	🛑	
└	_callOptionalReturn	Private 🔐	🛑	
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO❗
└	name	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	decimals	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	transfer	Public ❗	🛑	NO❗
└	allowance	Public ❗		NO❗

CONTRACT ASSESMENT

└	approve	Public ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	_transfer	Internal 🔒	🛑		
└	_mint	Internal 🔒	🛑		
└	_burn	Internal 🔒	🛑		
└	_approve	Internal 🔒	🛑		
└	_spendAllowance	Internal 🔒	🛑		
└	_beforeTokenTransfer	Internal 🔒	🛑		
└	_afterTokenTransfer	Internal 🔒	🛑		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	_checkOwner	Internal 🔒			
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑		
ReentrancyGuard	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
└	_nonReentrantBefore	Private 🔐	🛑		
└	_nonReentrantAfter	Private 🔐	🛑		
└	_reentrancyGuardEntered	Internal 🔒			
AlphaFinance	Implementation	ERC20, Ownable, ReentrancyGuard		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	<Fallback>	External ❗	💵	NO❗
└	getRouterAddress	Public ❗		NO❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO❗
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	isAutomatedMarketMakerPair	Public ❗		NO❗
└	toggleSwapBack	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑		
└	_alphaBack	Internal 🔒	🛑		
└	UnderLink	External ❗	🛑	NO❗

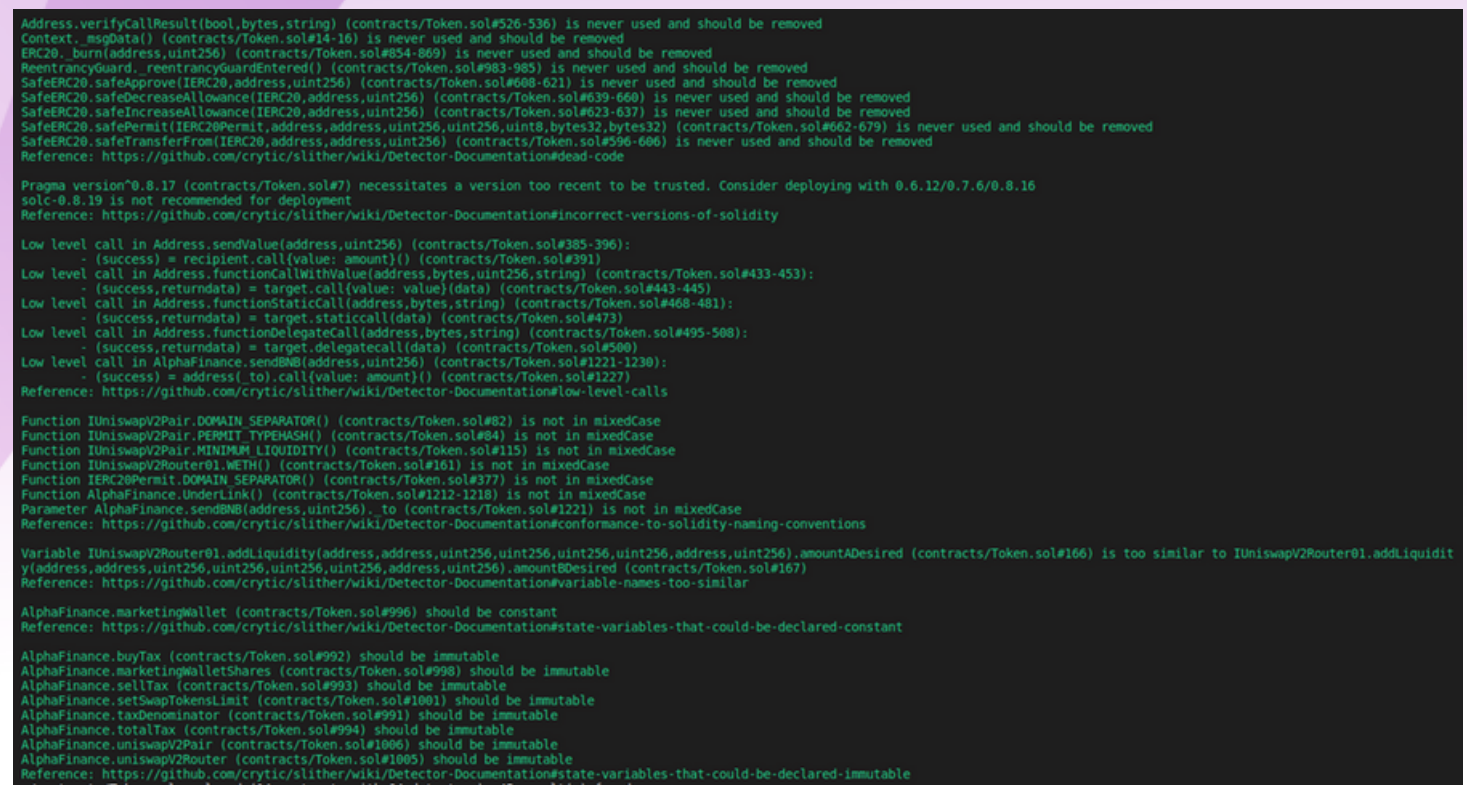
CONTRACT ASSESMENT

| └ | sendBNB | Internal 🔒 | 🛑 | nonReentrant |

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Address.verifyCallResult(bool,bytes,string) (contracts/Token.sol#526-536) is never used and should be removed
Context._msgData() (contracts/Token.sol#14-16) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#854-869) is never used and should be removed
ReentrancyGuard._reentrancyGuardEntered() (contracts/Token.sol#983-985) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/Token.sol#608-621) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/Token.sol#639-660) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/Token.sol#623-637) is never used and should be removed
SafeERC20.safePermit(IERC20Permit,address,address,uint256,uint256,uint8,bytes32,bytes32) (contracts/Token.sol#662-679) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/Token.sol#596-606) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/Token.sol#7) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.19 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#385-396):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#391)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/Token.sol#433-453):
        - (success,returndata) = target.call{value: value}(data) (contracts/Token.sol#443-445)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/Token.sol#468-481):
        - (success,returndata) = target.staticcall(data) (contracts/Token.sol#473)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/Token.sol#495-508):
        - (success,returndata) = target.delegatecall(data) (contracts/Token.sol#500)
Low level call in AlphaFinance.sendBNB(address,uint256) (contracts/Token.sol#1221-1230):
        - (success) = address(_to).call{value: amount}() (contracts/Token.sol#1227)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#82) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#84) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#115) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/Token.sol#161) is not in mixedCase
Function IERC20Permit.DOMAIN_SEPARATOR() (contracts/Token.sol#377) is not in mixedCase
Function AlphaFinance.UnderLink() (contracts/Token.sol#1212-1218) is not in mixedCase
Parameter AlphaFinance.sendBNB(address,uint256)._to (contracts/Token.sol#1221) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#166) is too similar to IUniswapV2Router01.addLiquidit
y(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#167)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

AlphaFinance.marketingWallet (contracts/Token.sol#996) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

AlphaFinance.buyTax (contracts/Token.sol#992) should be immutable
AlphaFinance.marketingWalletShares (contracts/Token.sol#998) should be immutable
AlphaFinance.sellTax (contracts/Token.sol#993) should be immutable
AlphaFinance.setSwapTokensLimit (contracts/Token.sol#1001) should be immutable
AlphaFinance.taxDenominator (contracts/Token.sol#991) should be immutable
AlphaFinance.totalTax (contracts/Token.sol#994) should be immutable
AlphaFinance.uniswapV2Pair (contracts/Token.sol#1006) should be immutable
AlphaFinance.uniswapV2Router (contracts/Token.sol#1005) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No issues found

FUNCTIONAL TESTING

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding Liquidity (Passed):
liquidity added on Pancakeswap V2:

https://testnet.bscscan.com/tx/0xc4aae1784625f5a39033458e3dc39f69110904156ea3189a2408d377b2c029ef

2- Buying when excluded (0%)(Passed):

https://testnet.bscscan.com/tx/0x0aabbc417ca7c263e00c4775182b06fafea3180bf57bfc188bf0513aa682e53f

3- Selling when excluded (0%)(Passed):

https://testnet.bscscan.com/tx/0x1d224fc135cb5dcf56a5df5b55c2ac843e4e4680fce8085ba8074ab3d5a20f21

4- Transferring when excluded (0% tax) (passed):

https://testnet.bscscan.com/tx/0xc755196d9ccb0ebaef8720b524df91b67fcd06c57cc769268f54de3e2f262aee

5- Buying when not excluded (0% tax) (passed):

https://testnet.bscscan.com/tx/0xf38e75db264fd8869b077f05157ab9380e25b32efccaeb9f16f4fb94d7b4f118

FUNCTIONAL TESTING

6- Selling when not excluded (1% tax) (passed):

https://testnet.bscscan.com/tx/0x0f9db5d9474dc5ad0ba313c9935a1166a1ea260e3ced1469f8875004792abd55

7- Transferring when not excluded(0% tax) (passed):

https://testnet.bscscan.com/tx/0xb23feb6cd5bf28c2f106cf40aa05db3a28242bf8c1402865309305b3b4a6b9e9

8- Internal swap (passed):

marketing wallet received BNB
https://testnet.bscscan.com/address/0xed172e1ef6426fc41aedecce4097a4391c7ae82a#internaltx

MANUAL TESTING

Informative – Non-authorized call to UnLink

Severity: **Low**

Function: UnderLink

Lines: 1212

Status: not resolved

Overview:

The UnderLink function can be called by anyone, rendering the swap threshold set by setSwapTokensAtAmount ineffective. This is because the _alphaBack function, which is called by UnderLink, will always swap the entire contract token balance, regardless of the set swap threshold.

```
function UnderLink() external {
    uint256 contractTokenBalance = balanceOf(address(this));

    require(contractTokenBalance > 0, "Cant Swap Back 0 Token!");

    _alphaBack(contractTokenBalance);
}
```

Recommendation:

• To address the swap threshold ineffectiveness issue, we recommend modifying the UnderLink function to swap tokens only if the contract token balance is greater than or equal to the set swap threshold:

```
function UnderLink() external {
  uint256 contractTokenBalance = balanceOf(address(this));

  require(contractTokenBalance >= swapTokensAtAmount, "Contract token balance must be greater than or equal to the swap threshold");

  _alphaBack(contractTokenBalance);
}
```

By implementing these changes, you can reduce gas usage and ensure that the swap threshold is effective in controlling the token swap frequency.

Informative – Redundant code

Severity: **Low**

Function: _alphaBack

Lines: 1201-1207

Status: not resolved

Overview:

The _alphaBack function calculates the marketing share of the ETH balance obtained from swapping tokens, and then sends the calculated marketing share to the marketing wallet. However, since marketingWalletShares is always set to 100%, the calculation of marketingShare becomes redundant and leads to some extra gas usage.

```
uint256 newBalance = address(this).balance - initialBalance;

uint256 marketingShare = (newBalance * marketingWalletShares) / 100;

if (marketingShare > 0) {
    sendBNB(marketingWallet, marketingShare);
}
```

Recommendation:

• Directly send all the ETH balance of contract to marketing wallet.

Informative – Redundant code

Severity: **Low**

Function: _transfer

Lines: 1179

Status: not resolved

Overview:

In the _transfer function, there is a possibility of a zero-value transfer when fees are equal to 0. Specifically, the line super._transfer(from, address(this), fees); can lead to a redundant zero-value transfer, resulting in unnecessary gas consumption.

```
if (takeFee) {
    uint256 fees = 0;
    if (_isAutomatedMarketMakerPair[from]) {
        fees = (amount * buyTax) / taxDenominator;
    } else if (_isAutomatedMarketMakerPair[to]) {
        fees = (amount * sellTax) / taxDenominator;
    }

    amount = amount - fees;

    super._transfer(from, address(this), fees); //@audit zero transfer, if fees == 0
}
```

Recommendation:

To optimize the _transfer function and eliminate the redundant zero-value transfer, we recommend adding a conditional check to ensure that fees are greater than 0 before executing the super._transfer(from, address(this), fees);

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace