Cyberscope

Audit Report

AlphaGate

October 2023

SHA256 77a1d9f2a32b4acdc78ed428a08831ff6a9ce511a9fdca3dba8e1e2194e430e9

Audited by © cyberscope

Analysis

Critical Medium Minor / Informative Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Unresolved
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	RCS	Redundant Conditional Statement	Unresolved
	MEE	Missing Events Emission	Unresolved
	PAV	Pair Address Validation	Unresolved
	IDI	Immutable Declaration Improvement	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L07	Missing Events Arithmetic	Unresolved

Table of Contents

Analysis **1**
Diagnostics **2**
Table of Contents **3**
Review **4**
Audit Updates 4
Source Files 4
Findings Breakdown **5**
ST - Stops Transactions 6
Description 6
Recommendation 6
RCS - Redundant Conditional Statement 7
Description 7
Recommendation 7
MEE - Missing Events Emission 8
Description 8
Recommendation 8
PAV - Pair Address Validation 9
Description 9
Recommendation 9
IDI - Immutable Declaration Improvement 10
Description 10
Recommendation 10
L04 - Conformance to Solidity Naming Conventions 11
Description 11
Recommendation 12
L07 - Missing Events Arithmetic 13
Description 13
Recommendation 13
Functions Analysis **14**
Inheritance Graph **18**
Flow Graph **19**
Summary **20**
Disclaimer **21**
About Cyberscope **22**

Review

Contract Name	AlphaGate
Symbol	AGATE
Decimals	9
Total Supply	100,000,000

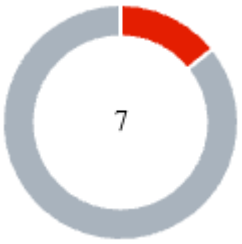
Audit Updates

Initial Audit	28 Oct 2023
Corrected Phase 2	31 Oct 2023

Source Files

Filename	SHA256
AlphaGate.sol	77a1d9f2a32b4acdc78ed428a08831ff6a9ce511a9fdca3dba8e1e2194e430e9

Findings Breakdown



Critical	1
Medium	0
Minor / Informative	6

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	0	0	0	0
Minor / Informative	6	0	0	0

ST - Stops Transactions

Criticality	Critical
Location	AlphaGate.sol#L475
Status	Unresolved

Description

The transactions are initially disabled for all users excluding the authorized addresses. The owner can enable the transactions for all users. Once the transactions are enabled the owner will not be able to disable them again.

```
if(!tradingOpen) {
    require(isFeeExempt[sender] || isFeeExempt[recipient], "Trading is
disabled");
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. Some suggestions are:

- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

RCS - Redundant Conditional Statement

Criticality	Minor / Informative
Location	AlphaGate.sol#L329
Status	Unresolved

Description

The contract includes the `setTxLimit` function, which is designed to set the `finalMaxTxLimitPercent` to a new value. However, there is a redundant condition within this function. Specifically, the condition `_newMaxTxLimitPercent < 50` is unnecessary, as the `initialMaxTxLimitPercent` is a constant variable set to 1. This means that the only valid value that can be assigned to finalMaxTxLimitPercent is 0, and the condition will always evaluate to true. Therefore, the condition is redundant and serves no practical purpose.

```
function setTxLimit(uint256 _newMaxTxLimitPercent) external onlyOwner {
    require(_newMaxTxLimitPercent < initialMaxTxLimitPercent &&
_newMaxTxLimitPercent < 50, "Transaction limit must be lower than the
initial transaction limit and lower than 50%");
    finalMaxTxLimitPercent = _newMaxTxLimitPercent;
}
```

Recommendation

The team is advised to remove the redundant condition `_newMaxTxLimitPercent < 50` from the `setTxLimit` function, as it does not impact the functionality of the contract and may only create confusion or unnecessary complexity. By simplifying the code, the contract becomes more understandable and less prone to potential misunderstandings. However, if there are different requirements or intentions, adjustments to the logic may be necessary to achieve the desired behavior. It's important to ensure that the code reflects the project's specific needs and objectives.

MEE - Missing Events Emission

Criticality	Minor / Informative
Location	AlphaGate.sol#L336
Status	Unresolved

Description

The contract performs actions and state mutations from external methods that do not result in the emission of events. Emitting events for significant actions is important as it allows external parties, such as wallets or dApps, to track and monitor the activity on the contract. Without these events, it may be difficult for external parties to accurately determine the current state of the contract.

```
isFeeExempt[addressToExempt] = isExempt;
```

Recommendation

It is recommended to include events in the code that are triggered each time a significant action is taking place within the contract. These events should include relevant details such as the user's address and the nature of the action taken. By doing so, the contract will be more transparent and easily auditable by external parties. It will also help prevent potential issues or disputes that may arise in the future.

PAV - Pair Address Validation

Criticality	Minor / Informative
Location	AlphaGate.sol#L339
Status	Unresolved

Description

The `setUniswapPair` function allows the contract owner to set the `uniswapPairAddress` to any arbitrary value without validation. This lack of validation can lead to unintended behavior, including the potential disruption of the contract's intended functionality.

```
function setUniswapV2Pair(address pairAddress) external onlyOwner {
    require(pairAddress != address(0), "ZeroAddress not allowed");
    require(!tradingOpen, "Trading is already open");
    uniswapV2PairAddress = pairAddress;
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

IDI - Immutable Declaration Improvement

Criticality	Minor / Informative
Location	AlphaGate.sol#L306
Status	Unresolved

Description

The contract declares state variables that their value is initialized once in the constructor and are not modified afterwards. The `immutable` is a special declaration for this kind of state variables that saves gas when it is defined.

```
taxAddress
```

Recommendation

By declaring a variable as immutable, the Solidity compiler is able to make certain optimizations. This can reduce the amount of storage and computation required by the contract, and make it more gas-efficient.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	AlphaGate.sol#L135,279,280,282,286,287,288,289,320,329,581,600
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
function WETH() external pure returns (address);
string private constant _name = "AlphaGate"
string private constant _symbol = "AGATE"
uint8 private constant _decimals = 9
uint256 private constant totalBuyTax = 4
uint256 private constant totalSellTax = 4
uint256 private constant totalBurnTax = 1
uint256 private constant initialMaxTxLimitPercent = 1
uint256 _newSwapThresholdPercent
uint256 _newMaxTxLimitPercent
uint256 _tokenAmount
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.
Find more information on the Solidity documentation
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L07 - Missing Events Arithmetic

Criticality	Minor / Informative
Location	AlphaGate.sol#L321,331
Status	Unresolved

Description

Events are a way to record and log information about changes or actions that occur within a contract. They are often used to notify external parties or clients about events that have occurred within the contract, such as the transfer of tokens or the completion of a task.

It's important to carefully design and implement the events in a contract, and to ensure that all required events are included. It's also a good idea to test the contract to ensure that all events are being properly triggered and logged.

```
swapThresholdPercent = _newSwapThresholdPercent
finalMaxTxLimitPercent = _newMaxTxLimitPercent
```

Recommendation

By including all required events in the contract and thoroughly testing the contract's functionality, the contract ensures that it performs as intended and does not have any missing events that could cause issues with its arithmetic.

Functions Analysis

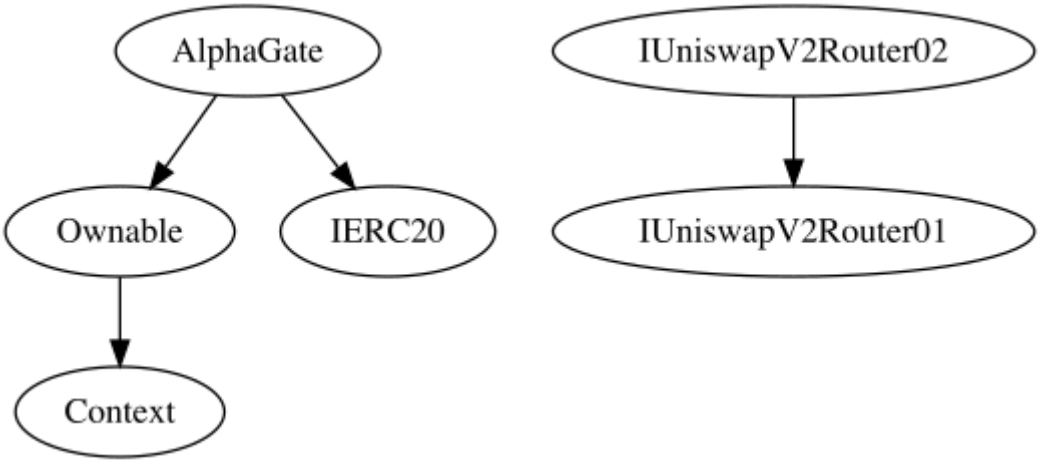
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-

	transferFrom	External	✓	-
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-

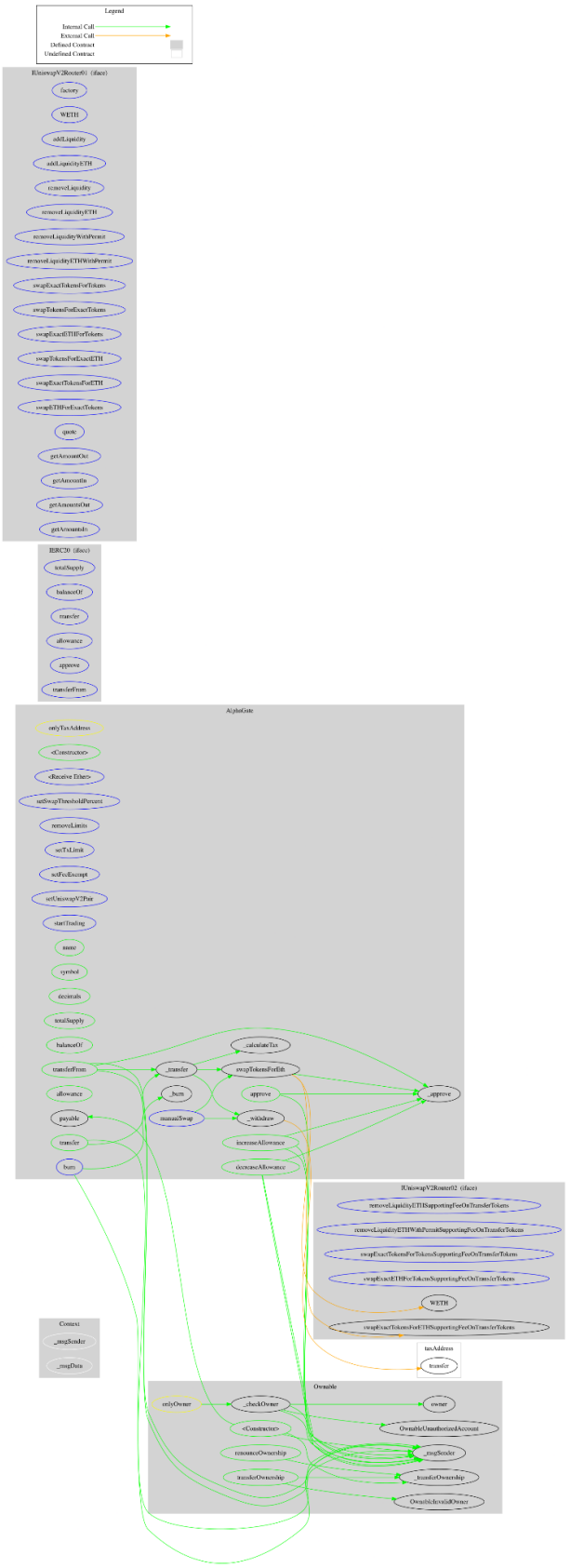
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
AlphaGate	Implementation	IERC20, Ownable		
		Public	✓	-
		External	Payable	-
	setSwapThresholdPercent	External	✓	onlyTaxAddress
	removeLimits	External	✓	onlyOwner
	setTxLimit	External	✓	onlyOwner
	setFeeExempt	External	✓	onlyOwner
	setUniswapV2Pair	External	✓	onlyOwner
	startTrading	External	✓	onlyOwner
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-

	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_calculateTax	Internal		
	_burn	Internal	✓	
	burn	External	✓	-
	swapTokensForEth	Internal	✓	
	_withdraw	Internal	✓	
	manualSwap	External	✓	onlyTaxAddress
	_approve	Internal	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IUniswapV2Router01 (iface)
factory
WETH
addLiquidity
addLiquidityETH
removeLiquidity
removeLiquidityETH
removeLiquidityWithPermit
removeLiquidityETHWithPermit
swapExactTokensForTokens
swapTokensForExactTokens
swapExactETHForTokens
swapTokensForExactETH
swapExactTokensForETH
swapETHForExactTokens
quote
getAmountOut
getAmountIn
getAmountsOut
getAmountsIn
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
AlphaGate
onlyTaxAddress
<Constructor>
<Receive Ether>
setSwapThresholdPercent
removeLimits
setTxLimit
setFeeExcept
setUniswapV2Pair
startTrading
name
symbol
decimals
totalSupply
balanceOf
transferFrom
allowance
payable
transfer
burn
_transfer
_burn
manualSwap
_calculateTax
swapTokensForEth
approve
_withdraw
increaseAllowance
decreaseAllowance
_approve
IUniswapV2Router02 (iface)
removeLiquidityETHSupportingFeeOnTransferTokens
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens
swapExactTokensForTokensSupportingFeeOnTransferTokens
swapExactETHForTokensSupportingFeeOnTransferTokens
WETH
swapExactTokensForETHSupportingFeeOnTransferTokens
Context
_msgSender
_msgData
taxAddress
transfer
Ownable
onlyOwner
_checkOwner
owner
<Constructor>
OwnableUnauthorizedAccount
renounceOwnership
_msgSender
transferOwnership
_transferOwnership
OwnableInvalidOwner

Summary

AlphaGate contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like stop transactions. A multi-wallet signing pattern will provide security against potential hacks. There is also a limit of max 5% fees buy and sell fees. Additionally, the contract implements a fee mechanism where that charges users with 25% buy fees and 35% sell fees for the first 20 transactions.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io