Audit Report

ACG

June 2023

Network	BSC
Address	0x41b296076F4432cF56cDc5598Ea29203f3a9B17a
Audited by	© cyberscope

Analysis

🔴 Critical 🟠 Medium ⚪ Minor / Informative 🔵 Pass

Severity	Code	Description	Status
🔵	ST	Stops Transactions	Passed
🔵	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
●	FRV	Fee Restoration Vulnerability	Unresolved
●	ILA	Invalid Liquidity Addition	Unresolved
●	MMN	Misleading Method Naming	Unresolved
●	L14	Uninitialized Variables in Local Scope	Unresolved
●	L18	Multiple Pragma Directives	Unresolved

Table of Contents

Review

Contract Name	ACG
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Explorer	https://bscscan.com/address/0x41b296076f4432cf56cdc5598ea29203f3a9b17a
Address	0x41b296076f4432cf56cdc5598ea29203f3a9b17a
Network	BSC
Symbol	ACG
Decimals	18
Total Supply	1.000.000.000

Audit Updates

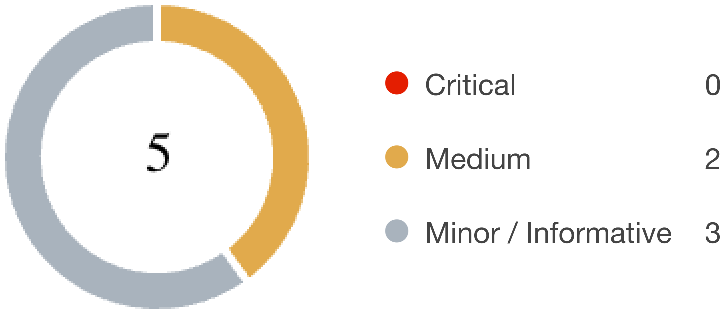
Initial Audit	23 May 2023 https://github.com/cyberscope-io/audits/blob/main/acg/v1/audit.pdf
Corrected Phase 2	28 May 2023 https://github.com/cyberscope-io/audits/blob/main/acg/v2/audit.pdf
Corrected Phase 3	05 Jun 2023 https://github.com/cyberscope-io/audits/blob/main/acg/v3/audit.pdf
Corrected Phase 4	09 Jun 2023

Source Files

Filename	SHA256
contracts/acg.sol	2e254f6a5db83a9bb2bbef5a67e515f85999cd58e9191a11074694cacf660321
contracts/contracts/access/Ownable.sol	42df7a70b8190e7c8e3aeb443aeacc2b23b389b18fa2ce00e9eb60a367a2bb20
contracts/contracts/interfaces/IUniswapV2Factory.sol	3dd4c1f051cee242d1c81b3868d19d983706f47dc6d4e61c83e8645dab7b190f
contracts/contracts/interfaces/IUniswapV2Pair.sol	d031a0cf0541e16cc08a0772453796dcbf77727976822ac038dbea47e16171cb
contracts/contracts/interfaces/IUniswapV2Router01.sol	9e9232b0ab8af12bf698a622047a0057ab2b5b068360e24c8599576a40653601
contracts/contracts/interfaces/IUniswapV2Router02.sol	add2f9ec336a24dfe0fcf25cd27fd11860fa09f8e303867f5188b2b1769b31e4
contracts/contracts/token/ERC20/ERC20.sol	bce14c3fd3b1a668529e375f6b70ffdf9cef8c4e410ae99608be5964d98fa701
contracts/contracts/token/ERC20/extensions/IERC20Metadata.sol	af5c8a77965cc82c33b7ff844deb9826166689e55dc037a7f2f790d057811990
contracts/contracts/token/ERC20/extensions/IERC20Permit.sol	2919f8aa74c48a2fc38fff7875ebc9d1604e9180f8c57416ba1ee589fe0dde60
contracts/contracts/token/ERC20/IERC20.sol	94f23e4af51a18c2269b355b8c7cf4db8003d075c9c541019eb8dcf4122864d5
contracts/contracts/token/ERC20/SafeERC20.sol	1d489ce3f5dd4966c090782c7547f51128968221f592301153c0644dfe862179
contracts/contracts/utils/Address.sol	8160a4242e8a7d487d940814e5279d934e81f0436689132a4e73394bab084a6d

contracts/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a

Findings Breakdown



Severity	Unresolved	Acknowledged	Resolved	Other
● Critical	0	0	0	0
● Medium	2	0	0	0
● Minor / Informative	3	0	0	0

FRV - Fee Restoration Vulnerability

Criticality	Medium
Location	contracts/acg.sol#L1063
Status	Unresolved

Description

The contract demonstrates a potential vulnerability upon removing and restoring the fees. This vulnerability can occur when the fees have been set to zero. During a transaction, if the fees have been set to zero, then both remove fees and restore fees functions will be executed. The remove fees function is executed to temporarily remove the fees, ensuring the sender is not taxed during the transfer. However, the function prematurely returns without setting the variables that hold the previous fee values.

As a result, when the subsequent restore fees function is called after the transfer, it restores the fees to their previous values. However, since the previous fee values were not properly set to zero, there is a risk that the fees will retain their non-zero values from before the fees were removed. This can lead to unintended consequences, potentially causing incorrect fee calculations or unexpected behavior within the contract.

```
   function _removeAllFees() private {
        if (
            operationFee == 0 &&
            ...
        ) return;

        _previousOperationFee = operationFee;
        ...

        operationFee = 0;
        ...
    }

    function _restoreAllFees() private {
        operationFee = _previousOperationFee;
        ...
    }
```

Recommendation

The team is advised to modify the remove fees function to ensure that the previous fee values are correctly set to zero, regardless of their initial values. A recommended approach would be to remove the early return when both fees are zero.

ILA - Invalid Liquidity Addition

Criticality	Medium
Location	contracts/acg.sol#L1758
Status	Unresolved

Description

The contract adds liquidity to the pair as part of the liquidation process. The contract supports 2 different ways to add liquidity. The first one is by adding liquidity to the $token - BNB pair. The second one transfers $token - $otherToken to the pair. The process of transferring tokens to the pair address does not add liquidity. This method may create wrong assumptions that liquidity has been added.

```
function _transferToPair(
    address tokenAddress,
    address pair,
    uint256 otherHalf,
    uint256 newBalance
) private {
    IERC20(address(this)).safeTransfer(pair, otherHalf);
    IERC20(tokenAddress).safeTransfer(pair, newBalance);
}
```

Recommendation

The team is advised to properly add liquidity to the tokens pair address. This can be achieved by using the router's method `.addLiquidity()` .

MMN - Misleading Method Naming

Criticality	Minor / Informative
Location	contracts/acg.sol#L1702
Status	Unresolved

Description

Methods can have misleading names if their names do not accurately reflect the functionality they contain or the purpose they serve. The contract uses some method names that are too generic or do not clearly convey the underneath functionality. Misleading method names can lead to confusion, making the code more difficult to read and understand. Methods can have misleading names if their names do not accurately reflect the functionality they contain or the purpose they serve. The contract uses some method names that are too generic or do not clearly convey the underneath functionality. Misleading method names can lead to confusion, making the code more difficult to read and understand.

The method name `_pairValidation()` implies that it validates if the pair address is valid. On the contrary, this method returns the proper pair address according to the `pairToUse` property.

```
function _pairValidation() internal view returns (address, address,
bool) {}
```

Recommendation

It's always a good practice for the contract to contain method names that are specific and descriptive. The team is advised to keep in mind the readability of the code.

L14 - Uninitialized Variables in Local Scope

Criticality	Minor / Informative
Location	contracts/acg.sol#L1703,1704
Status	Unresolved

Description

Using an uninitialized local variable can lead to unpredictable behavior and potentially cause errors in the contract. It's important to always initialize local variables with appropriate values before using them.

```
address pair
address tokenAddress
```

Recommendation

By initializing local variables before using them, the contract ensures that the functions behave as expected and avoid potential issues.

L18 - Multiple Pragma Directives

Criticality	Minor / Informative
Location	contracts/acg.sol
Status	Unresolved

Description

If the contract includes multiple conflicting pragma directives, it may produce unexpected errors. To avoid this, it's important to include the correct pragma directive at the top of the contract and to ensure that it is the only pragma directive included in the contract.

```
pragma solidity ^0.8.1;
```

Recommendation

It is important to include only one pragma directive at the top of the contract and to ensure that it accurately reflects the version of Solidity that the contract is written in.

By including all required compiler options and flags in a single pragma directive, the potential conflicts could be avoided and ensure that the contract can be compiled correctly.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
ACG	Implementation	ERC20, Ownable		
		Public	✓	ERC20
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
		External	Payable	-
	_approve	Internal	✓	
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	getBNBBalance	Public		-
	getErc20TokenFeeBalance	Public		-
	getErc20TokensBalance	Public		-
	_checkTokenWBNBPair	Internal		
	_getTokenPair	Internal		
	setUSDTAddressAndPair	External	✓	onlyOwner

	setSpecialAddressAndPair	External	✓	onlyOwner
	setPairToUse	External	✓	onlyOwner
	setSwapRouter	External	✓	onlyOwner
	setSwapThresholdLimit	External	✓	onlyOwner
	setPenaltyTxAmount	External	✓	onlyOwner
	setErc20TokenAddress	External	✓	onlyOwner
	setAntiBotStatus	External	✓	onlyOwner
	setMaxBotSellCount	External	✓	onlyOwner
	setBotSellTimeLimit	External	✓	onlyOwner
	setOperationWallet	External	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setPoolsLeaderboardWallet	External	✓	onlyOwner
	setCommunityWallet	External	✓	onlyOwner
	setTreasuryOneWallet	External	✓	onlyOwner
	setTreasuryTwoWallet	External	✓	onlyOwner
	setPenaltyWallet	External	✓	onlyOwner
	_updateFees	Private	✓	
	_calculateTotalNewFees	Internal		
	setOperationFeePercent	External	✓	onlyOwner feesNotBeingSet
	setMarketingFeePercent	External	✓	onlyOwner feesNotBeingSet
	setLiquidityFeePercent	External	✓	onlyOwner feesNotBeingSet

	setPoolsLeaderboardFeePercent	External	✓	onlyOwner feesNotBeingSet
	setCommunityFeePercent	External	✓	onlyOwner feesNotBeingSet
	setTreasuryOneFeePercent	External	✓	onlyOwner feesNotBeingSet
	setTreasuryTwoFeePercent	External	✓	onlyOwner feesNotBeingSet
	setPenaltyFeePercent	External	✓	onlyOwner feesNotBeingSet
	_removeAllFees	Private	✓	
	_restoreAllFees	Private	✓	
	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	setSwapAndLiquifyEnabled	Public	✓	onlyOwner
	_calculateFees	Private		
	_calculateBuyFee	Private		
	_calculateSellFee	Private		
	_calculateFee	Private		
	_getCurrentSupply	Private		
	_getRate	Private		
	tokenFromReflection	Public		-
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		

	_getRValues	Private		
	_calculateERC20TokenFees	Private		
	_calculateLiquidityTokenFees	Private		
	_takeFees	Private	✓	
	_beforeTokenTransfer	Internal		
	_checkCanTransfer	Internal		
	_checkAntibotStatus	Internal		
	_transfer	Internal	✓	
	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_reflectBot	Private	✓	
	_swapAndGetFees	Private	✓	lockTheSwap
	_swapAndLiquify	Private	✓	
	_checkPairAndSwap	Private	✓	
	_pairValidation	Internal		
	_swapAndTransferToPair	Private	✓	
	_transferToPair	Private	✓	
	_calculateAvailableFeesAndTransfer	Private	✓	
	_transferFeesToWallet	Private	✓	
	_swapTokensForBnb	Private	✓	
	_addLiquidity	Private	✓	
	_swapTokensForTokens	Private	✓	
	manualBNBSwap	External	✓	onlyOwner

	manualERC20Swap	External	✓	onlyOwner lockTheSwap
	autoERC20Swap	External	✓	onlyOwner
	recoverBNB	External	✓	onlyOwner
	recoverBNBToWallet	External	✓	onlyOwner
	recoverERC20Tokens	External	✓	onlyOwner
	recoverERC20TokensToWallet	External	✓	onlyOwner
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-

IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-

	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	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		-

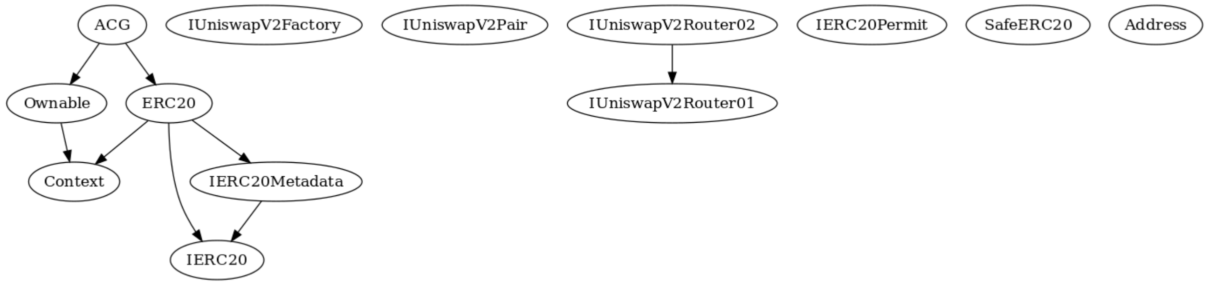
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
ERC20	Implementation	Context, IERC20, IERC20Metadata		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-

	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
IERC20Permit	Interface			
	permit	External	✓	-
	nonces	External		-
	DOMAIN_SEPARATOR	External		-
IERC20	Interface			

	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	forceApprove	Internal	✓	
	safePermit	Internal	✓	
	_callOptionalReturn	Private	✓	
	_callOptionalReturnBool	Private	✓	
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	

	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResultFromTarget	Internal		
	verifyCallResult	Internal		
	_revert	Private		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		

Inheritance Graph



Flow Graph

Summary

ACG contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. ACG is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The Contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions. There is also a limit of max 15% buy fees and 25% if the transferred amount is a specific threshold that is defined by the contract owner. Lastly, the contract has an antibot throttling mechanism that can prevent the transfers of up to 100 blocks.

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