Cyberscope

Audit Report

Catpurr

February 2024

Network	BSC
Address	0x20bc80955b3893b012bc0fba3d1605de57e00c1c
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
●	RSW	Redundant Storage Writes	Acknowledged
●	L07	Missing Events Arithmetic	Acknowledged
●	L07	Missing Events Arithmetic	Acknowledged
●	L16	Validate Variable Setters	Acknowledged
●	L19	Stable Compiler Version	Acknowledged

Table of Contents

Review

Contract Name	CATPURR
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Explorer	https://bscscan.com/address/0x20bc80955b3893b012bc0fba3d1605de57e00c1c
Address	0x20bc80955b3893b012bc0fba3d1605de57e00c1c
Network	BSC
Symbol	PURR
Decimals	18
Total Supply	1,000,000,000,000
Badge Eligibility	Yes

Contract Name	ExponentialTaxHandler
Explorer	https://bscscan.com/address/0x47Bf354F148D732C8145FaE4e3440371eB1902A2

Contract Name	TreasuryHandler
Explorer	https://bscscan.com/address/0x30671F6014eE389578356aa4568905b1f1cED593

Audit Updates

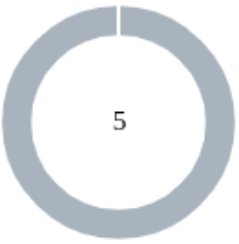
Initial Audit	19 Feb 2024 https://github.com/cyberscope-io/audits/blob/main/2-purr/v1/audit.pdf
Corrected Phase 2	21 Feb 2024 https://github.com/cyberscope-io/audits/blob/main/2-purr/v2/audit.pdf
Corrected Phase 3	27 Feb 2024

Source Files

Filename	SHA256
contracts/Catpurr.sol	a81f3461e46cd9071281b9a1fc023d3b5aa1097f92db83860b14c94eed5d12aa
contracts/treasury/ITreasuryHandler.sol	d802cb3e29064191f1e18f02220cc8181a09b4dda9aa4785385507808d85713d
contracts/tax/ITaxHandler.sol	1861fb4ec6daa61d4d7f99cbe426e0a53f655b45b38352a0db9c35a64fe27883
contracts/interfaces/IERC20Burnable.sol	7d8240509cd52f429bc723fb9da3729cb3cb5bb8b396d5428fe4999cff561dab
@openzeppelin/contracts/utils/Context.sol	b2cfee351bcafd0f8f27c72d76c054df9b571b62cfac4781ed12c86354e2a56c
@openzeppelin/contracts/token/ERC20/IERC20.sol	7ebde70853ccafcf1876900dad458f46eb9444d591d39bfc58e952e2582f5587
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	af5c8a77965cc82c33b7ff844deb9826166689e55dc037a7f2f790d057811990

@openzeppelin/contracts/access/Ownable.sol	a8e4e1ae19d9bd3e8b0a6d46577eec098c 01fbaffd3ec1252fd20d799e73393b

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	5

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	0	5	0	0

RSW - Redundant Storage Writes

Criticality	Minor / Informative
Location	contracts/tax/ExponentialTaxHandler.sol#L42,48
Status	Acknowledged

Description

The contract modifies the state of the following variables without checking if their current value is the same as the one given as an argument. As a result, the contract performs redundant storage writes, when the provided parameter matches the current state of the variables, leading to unnecessary gas consumption and inefficiencies in contract execution.

```
    function addExempt(address exemption) external onlyOwner {
        if (_exempted.add(exemption)) {
            emit TaxExemptionUpdated(exemption, true);
        }
    }

    function removeExempt(address exemption) external onlyOwner
{
        if (_exempted.remove(exemption)) {
            emit TaxExemptionUpdated(exemption, false);
        }
    }
```

Recommendation

The team is advised to implement additional checks within to prevent redundant storage writes when the provided argument matches the current state of the variables. By incorporating statements to compare the new values with the existing values before proceeding with any state modification, the contract can avoid unnecessary storage operations, thereby optimizing gas usage.

L07 - Missing Events Arithmetic

Criticality	Minor / Informative
Location	contracts/tax/ExponentialTaxHandler.sol#L35
Status	Acknowledged

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.

```
_taxRate = taxRate
```

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.

L07 - Missing Events Arithmetic

Criticality	Minor / Informative
Location	contracts/tax/ExponentialTaxHandler.sol#L35
Status	Acknowledged

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.

```
_taxRate = taxRate
```

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.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	contracts/treasury/TreasuryHandler.sol#L19
Status	Acknowledged

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
treasury = payable(_treasury)
```

Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	contracts/Catpurr.sol#L2 contracts/tax/ExponentialTaxHandler.sol#L2 contracts/treasury/TreasuryHandler.sol#L2 contracts/treasury/ITreasuryHandler.sol#L2
Status	Acknowledged

Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.17;
```

Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

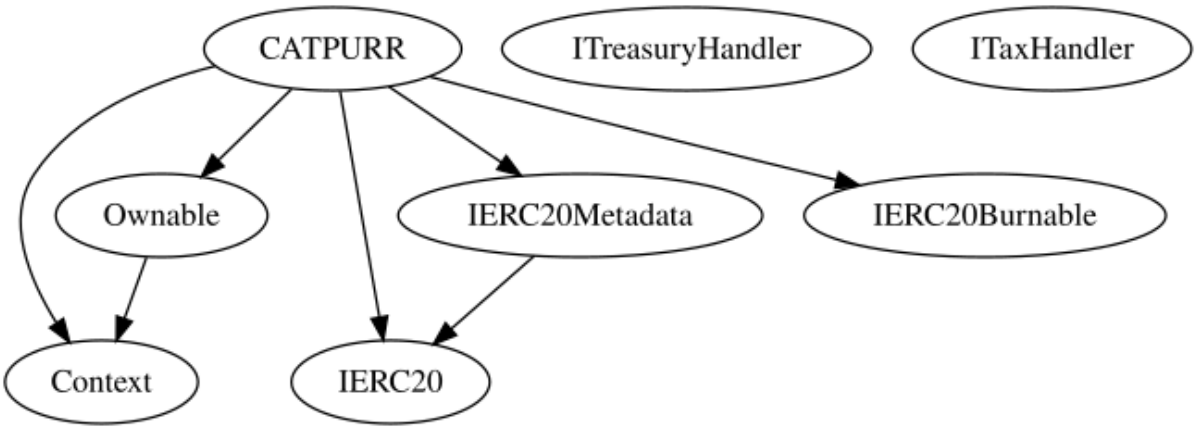
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
CATPURR	Implementation	Context, IERC20, IERC20Metadata, Ownable, IERC20Burnable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	setTaxHandler	External	✓	onlyOwner
	setTreasuryHandler	External	✓	onlyOwner
	burn	Public	✓	-
	burnFrom	Public	✓	-

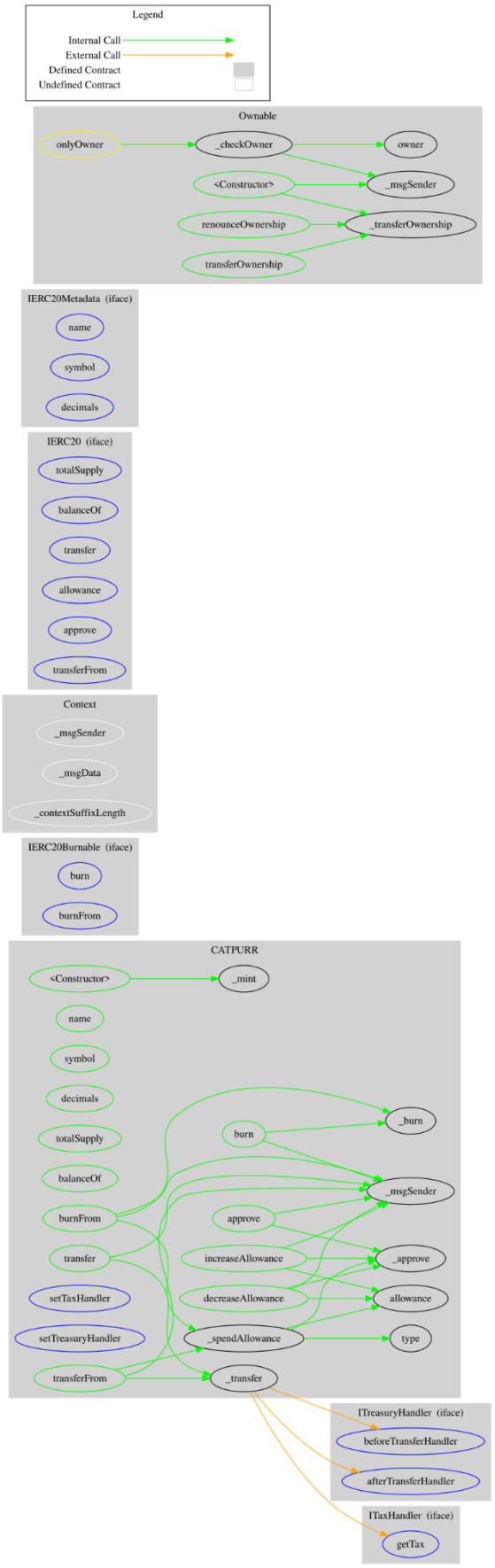
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
ITreasuryHandler	Interface			
	beforeTransferHandler	External	✓	-
	afterTransferHandler	External	✓	-
ITaxHandler	Interface			
	getTax	External		-
IERC20Burnable	Interface			
	burn	External	✓	-
	burnFrom	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
	_contextSuffixLength	Internal		
IERC20	Interface			

	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Ownable
onlyOwner
_checkOwner
owner
<Constructor>
_msgSender
renounceOwnership
_transferOwnership
transferOwnership
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
_msgData
_contextSuffixLength
IERC20Burnable (iface)
burn
burnFrom
CATPURR
<Constructor>
_mint
name
symbol
decimals
totalSupply
balanceOf
burnFrom
transfer
setTaxHandler
setTreasuryHandler
transferFrom
burn
approve
increaseAllowance
decreaseAllowance
_spendAllowance
_transfer
_burn
_msgSender
_approve
allowance
type
ITreasuryHandler (iface)
beforeTransferHandler
afterTransferHandler
ITaxHandler (iface)
getTax

Summary

Catpurr contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. Catpurr 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 25% fees.

The Catpurr's contract ownership has been renounced. The information regarding the transaction can be accessed through the following link:

https://bscscan.com/tx/0x3063822e1a77bba3d33b5586606f7924f6c16b166501bef52d1561fcfd7e16f4#eventlog

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