Cyberscope

Audit Report

Quantify

November 2024

Network BSC TESTNET

Address 0xA3242117A3087C47B36483506356Fbd1314dC6cD

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
	CCR	Contract Centralization Risk	Unresolved
	DTA	Dynamic Threshold Adjustment	Unresolved
	MEM	Misleading Error Messages	Unresolved
	PAMAR	Pair Address Max Amount Restriction	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved

Table of Contents

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation**: This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation**: This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical**: Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium**: Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor**: Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative**: Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
Critical	Highly Likely / High Impact
Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
Minor / Informative	Unlikely / Low to no Impact

Review

Contract Name	Quantify
Compiler Version	v0.8.4+commit.c7e474f2
Optimization	200 runs
Explorer	https://testnet.bscscan.com/address/0xa3242117a3087c47b36483506356fbd1314dc6cd
Address	0xa3242117a3087c47b36483506356fbd1314dc6cd
Network	BSC TESTNET
Symbol	QFI
Decimals	18
Total Supply	100,000,000
Badge Eligibility	Yes

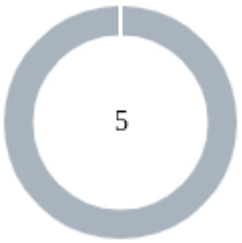
Audit Updates

Initial Audit	12 Nov 2024

Source Files

Filename	SHA256
Quantify.sol	e09da5a0bdddcd3e7fe4e116e142fcbc5e892749c6130e6ff31553d9eb389158

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	5	0	0	0

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	Quantify.sol#L483,496,505
Status	Unresolved

Description

The contract's functionality and behavior are heavily dependent on external parameters or configurations. While external configuration can offer flexibility, it also poses several centralization risks that warrant attention. Centralization risks arising from the dependence on external configuration include Single Point of Control, Vulnerability to Attacks, Operational Delays, Trust Dependencies, and Decentralization Erosion.

```
function changeAutoburnTax(uint8 newAutoburnTax) public onlyOwner
{}
function excludeFromTax(address user, bool exclude) public
onlyOwner {}
function excludeFromAntiwhale(address user, bool exclude) public
onlyOwner {}
function activateAntiwhale(bool activate) public onlyOwner {
```

Recommendation

To address this finding and mitigate centralization risks, it is recommended to evaluate the feasibility of migrating critical configurations and functionality into the contract's codebase itself. This approach would reduce external dependencies and enhance the contract's self-sufficiency. It is essential to carefully weigh the trade-offs between external configuration flexibility and the risks associated with centralization.

DTA - Dynamic Threshold Adjustment

Criticality	Minor / Informative
Location	Quantify.sol#L527,596
Status	Unresolved

Description

The contract enforces restrictions on the maximum balance of addresses. When an account's balance exceeds a specified `maxOwnable` threshold, the account is classified as a "whale" and is prohibited from receiving additional tokens.

Furthermore, the `maxOwnable` amount is dynamically adjusted through the `_burn` function each time the `autoburnTax` is applied. In such instances, the threshold for an account to be considered a "whale" is reduced proportionally to the total supply. Consequently, accounts that were not previously classified as "whales" may obtain this status and have the corresponding restrictions applied to them.

```
function _burn(address account, uint256 amount) internal virtual {
require(account != address(0), "ERC20: burn from the zero
address");
uint256 accountBalance = _balances[account];
require(accountBalance >= amount, "ERC20: burn amount exceeds
balance");
unchecked {
    _balances[account] = accountBalance - amount;
}
_totalSupply -= amount;
maxOwnable = _totalSupply * MAXOWNABLEPERCENTAGE / 100;
emit Transfer(account, address(0), amount);
}
```

Recommendation

The team is advised to ensure the current implementation aligns with the intended behaviour during code execution.

MEM - Misleading Error Messages

Criticality	Minor / Informative
Location	Quantify.sol#L562
Status	Unresolved

Description

The contract is using misleading error messages. Specifically, there are no error messages that accurately reflect the problem, making it difficult to identify and fix the issue. As a result, the users will not be able to find the root cause of the error.

```
uint constant MAXOWNABLEPERCENTAGE = 3;
```

```
require(_balances[to] + amount <= maxOwnable, "you can't own more than 5% of the total supply");}
```

Recommendation

The team is suggested to provide a descriptive message to the errors. This message can be used to provide additional context about the error that occurred or to explain why the contract execution was halted. This can be useful for debugging and for providing more information to users that interact with the contract.

PAMAR - Pair Address Max Amount Restriction

Criticality	Minor / Informative
Location	Quantify.sol#L527,562
Status	Unresolved

Description

The contract is configured to enforce a maximum token accumulation limit through checks. This mechanism aims to prevent excessive token concentration by reverting transactions that overcome the specified cap. However, this functionality encounters issues when transactions default to the pair address during sales. If the pair address is not listed in the exceptions, then the sale transactions are inadvertently stopped, effectively disrupting operations and making the contract susceptible to unintended behaviors akin to a honeypot.

```
function _antiwhalecheck(address to, uint amount) internal view {
if (!isExcludedFromAntiwhale[to] && antiwhale){
require(_balances[to] + amount <= maxOwnable, "you can't own more than 5% of
the total supply");}
}
```

Recommendation

It is advised to modify the contract to ensure uninterrupted operations by either permitting the pair address to exceed the established token accumulation limit or by safeguarding its status in the exception list. By recognizing and allowing these essential addresses the flexibility to hold more tokens than typical limits, the contract can maintain seamless transaction flows and uphold the liquidity and stability of the ecosystem. This modification is vital for avoiding disruptions that could impact the functionality and security of the contract.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	Quantify.sol#L471
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.

```
event autoburnTaxChanged(uint newAutoburnTax);
```

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/stable/style-guide.html#naming-conventions.

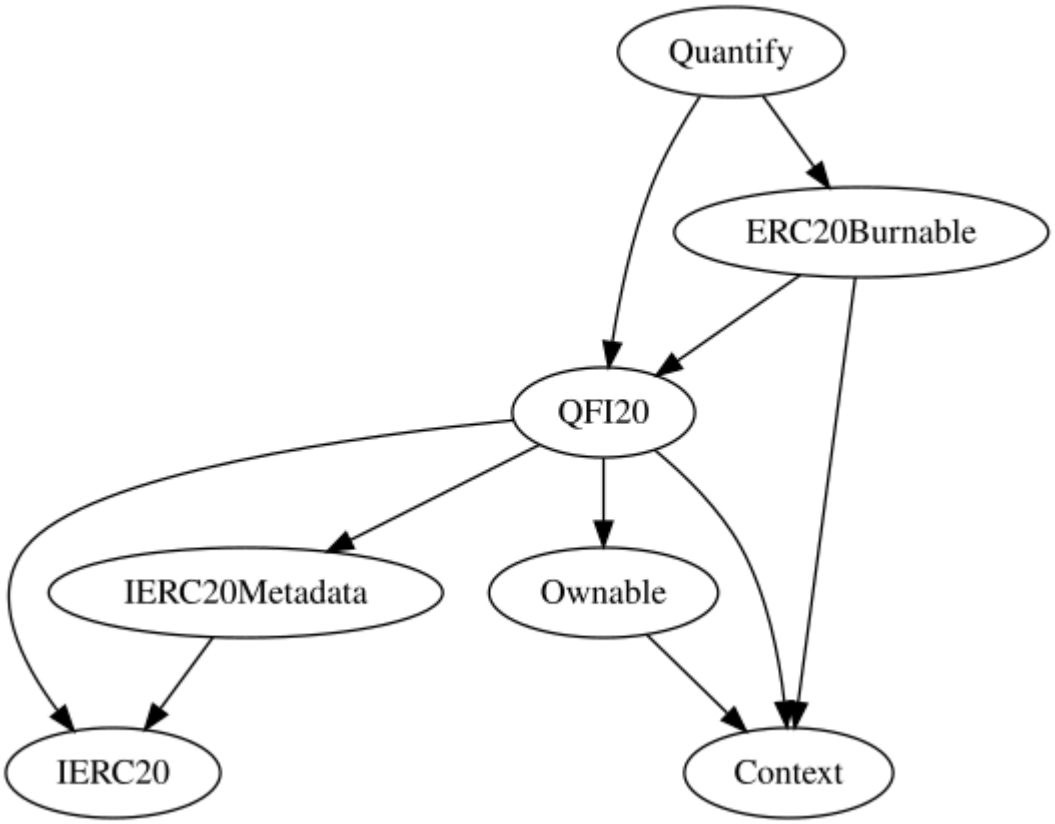
Functions Analysis

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	✓	-

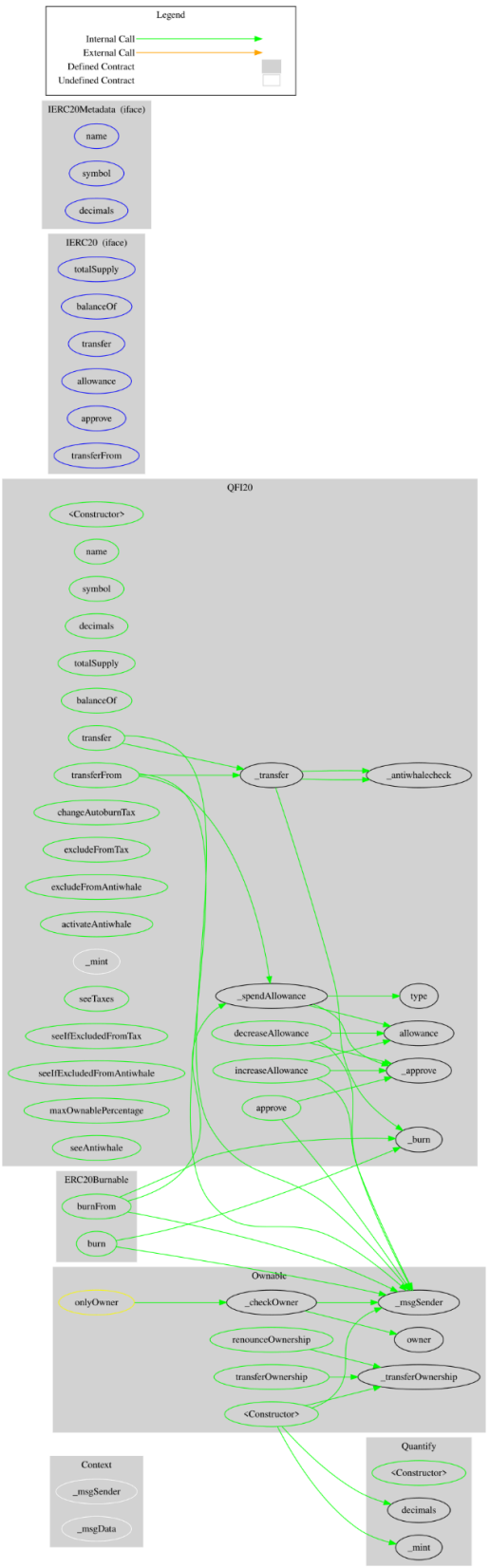
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
QFI20	Implementation	Context, IERC20, IERC20Metadata, Ownable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	changeAutoburnTax	Public	✓	onlyOwner
	excludeFromTax	Public	✓	onlyOwner
	excludeFromAntiwhale	Public	✓	onlyOwner
	activateAntiwhale	Public	✓	onlyOwner

	_transfer	Internal	✓	
	_antiwhalecheck	Internal		
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	seeTaxes	Public		-
	seeIfExcludedFromTax	Public		-
	seeIfExcludedFromAntiwhale	Public		-
	maxOwnablePercentage	Public		-
	seeAntiwhale	Public		-
ERC20Burnable	Implementation	Context, QFI20		
	burn	Public	✓	-
	burnFrom	Public	✓	-
Quantify	Implementation	QFI20, ERC20Burnable		
		Public	✓	QFI20

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
QFI20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
transfer
transferFrom
changeAutoburnTax
excludeFromTax
excludeFromAntiwhale
activateAntiwhale
_mint
seeTaxes
seeIfExcludedFromTax
seeIfExcludedFromAntiwhale
maxOwnablePercentage
seeAntiwhale
_transfer
_antiwhalecheck
_spendAllowance
decreaseAllowance
increaseAllowance
approve
type
allowance
_approve
_burn
ERC20Burnable
burnFrom
burn
Ownable
onlyOwner
_checkOwner
renounceOwnership
transferOwnership
<Constructor>
_msgSender
owner
_transferOwnership
Context
_msgSender
_msgData
Quantify
<Constructor>
decimals
_mint

Summary

Quantify contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. The Smart Contract analysis reported no compiler error or critical issues. There is also a limit of max 2% fees.

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

cyberscope.io