Cyberscope

Audit Report

Quickswap

February 2024

Network	Matic
Address	0xb5c064f955d8e7f38fe0460c556a72987494ee17
Audited by	© cyberscope

Table of Contents

Review

Contract Name	Quick
Compiler Version	v0.5.16+commit.9c3226ce
Optimization	1000000 runs
Explorer	https://polygonscan.com/address/0xb5c064f955d8e7f38fe0460c556a72987494ee17
Address	0xb5c064f955d8e7f38fe0460c556a72987494ee17
Network	MATIC
Symbol	QUICK
Decimals	18
Total Supply	883,439,539.725

Audit Updates

Initial Audit	12 Feb 2024

Source Files

Filename	SHA256
Quick.sol	f25f546d694ef6ac2e2c5d4bbaf33bd7b9364658c2901e3be48a0f4834fcbc4d

Overview

Quick Contract Overview

This document presents the overview of the smart contract audit conducted for the `Quick` contract of "Quickswap" project on the Polygon blockchain. The purpose of this audit is to identify and address security vulnerabilities, provide recommendations for code improvements, and ensure the robustness of the codebase.

Deposits and Withdrawals

Features token minting and burning, directly impacting the total supply. The `gateway` contract is responsible for the minting process.

Governance and Voting

- Incorporates delegated voting, allowing token holders to delegate their voting rights.
- Utilizes checkpoints to record and manage vote balances over time, ensuring accurate and historical vote power tracking.

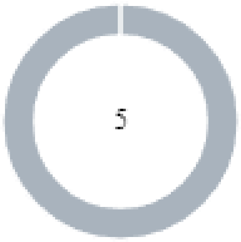
Token Transfer and Allowance

- Supports standard `transfer` and `transferFrom` functions for token movements.
- Implements `approve` and `permit` methods for managing allowances.

Event Logging

Incorporates events for tracking delegate changes and vote balance updates, enhancing transparency and auditability of governance actions.

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

Diagnostics

🔴 Critical 🟠 Medium ● Minor / Informative

Severity	Code	Description	Status
●	CCR	Contract Centralization Risk	Unresolved
●	DVPUI	Delegated Voting Power Update Inadequacy	Unresolved
●	L16	Validate Variable Setters	Unresolved
●	L17	Usage of Solidity Assembly	Unresolved
●	L19	Stable Compiler Version	Unresolved

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	Quick.sol#L267,278
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.

Specifically, the contract designates a `gateway` address, which is set during contract initialization and is responsible for token minting. This `gateway` is a proxy contract, implying that its underlying logic and address can be altered. While this design provides adaptability and the potential for upgrading contract functionalities, it inherently centralizes control to the address holding the capability to update the `gateway`. Additionally, given that the `gateway` contract has the authority to mint tokens, by calling the `deposit` function, this centralization places significant trust in the gateway's security.

```
constructor(address gateway_) public {

    gateway = gateway_;
}

function deposit(address user, bytes calldata depositData)
    external
{
    require(msg.sender == gateway,  "Invalid access");
    uint256 rawAmount = abi.decode(depositData, (uint256));
    uint96 amount = safe96(rawAmount, "Quick::deposit: amount
exceeds 96 bits");

    _mint(user, amount);
}
```

Recommendation

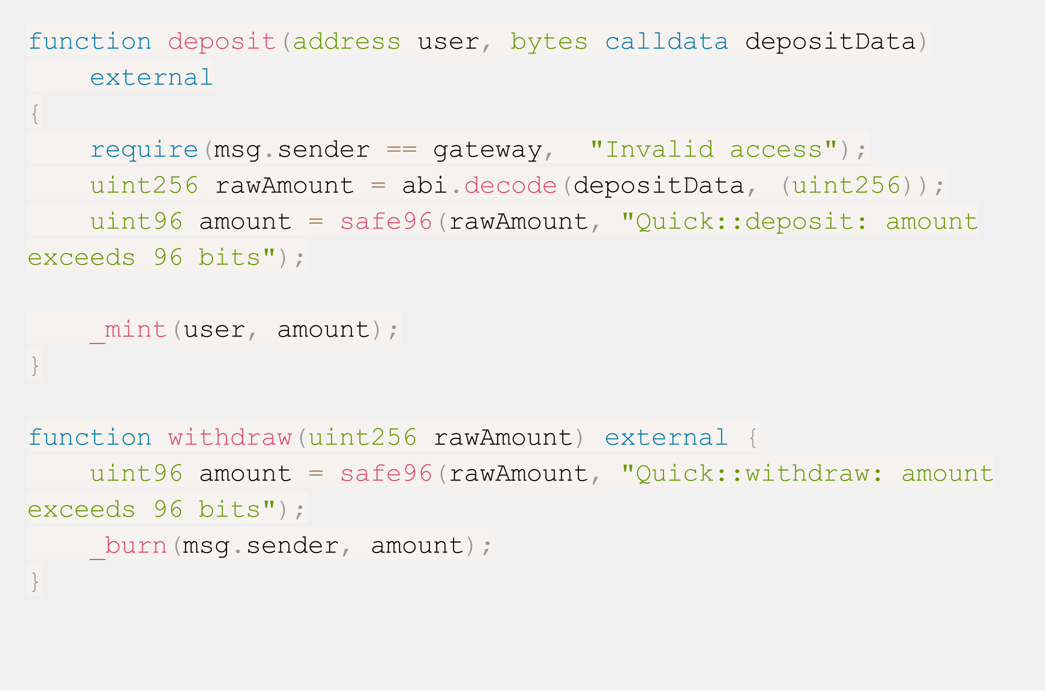
To address this finding and mitigate centralization risks, it is recommended to evaluate the feasibility of migrating critical configurations and functionality into the contract's codebase itself and re-evaluate the dependence on external configurations, particularly the `gateway` contract. This approach 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.

DVPUI - Delegated Voting Power Update Inadequacy

Criticality	Minor / Informative
Location	Quick.sol#L278,293
Status	Unresolved

Description

The `deposit` function is designed to mint tokens for users upon depositing on the root chain, while the `withdraw` function burns tokens when users wish to withdraw. However, neither function includes logic to update the delegates' voting power based on the change in token balances. This omission could lead to discrepancies between the actual token balances held by users and the voting power delegated on their behalf. As a result, voting power may not accurately reflect the current distribution of tokens among holders.



```
function deposit(address user, bytes calldata depositData)
    external
{
    require(msg.sender == gateway,  "Invalid access");
    uint256 rawAmount = abi.decode(depositData, (uint256));
    uint96 amount = safe96(rawAmount, "Quick::deposit: amount
exceeds 96 bits");

    _mint(user, amount);
}

function withdraw(uint256 rawAmount) external {
    uint96 amount = safe96(rawAmount, "Quick::withdraw: amount
exceeds 96 bits");
    _burn(msg.sender, amount);
}
```

Recommendation

It is recommended to revise the logic behind the adjustment of delegated voting power within both the `deposit` and `withdraw` functions. This involves ensuring that any change in token balances, whether through depositing or withdrawing, is accurately mirrored in the voting power attributed to token holders and their delegates.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	Quick.sol#L267
Status	Unresolved

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.

```
gateway = gateway_
```

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.

L17 - Usage of Solidity Assembly

Criticality	Minor / Informative
Location	Quick.sol#L559
Status	Unresolved

Description

Using assembly can be useful for optimizing code, but it can also be error-prone. It's important to carefully test and debug assembly code to ensure that it is correct and does not contain any errors.

Some common types of errors that can occur when using assembly in Solidity include Syntax, Type, Out-of-bounds, Stack, and Revert.

```
assembly { chainId := chainid() }
```

Recommendation

It is recommended to use assembly sparingly and only when necessary, as it can be difficult to read and understand compared to Solidity code.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	Quick.sol#L13
Status	Unresolved

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 ^0.5.16;
```

Recommendation

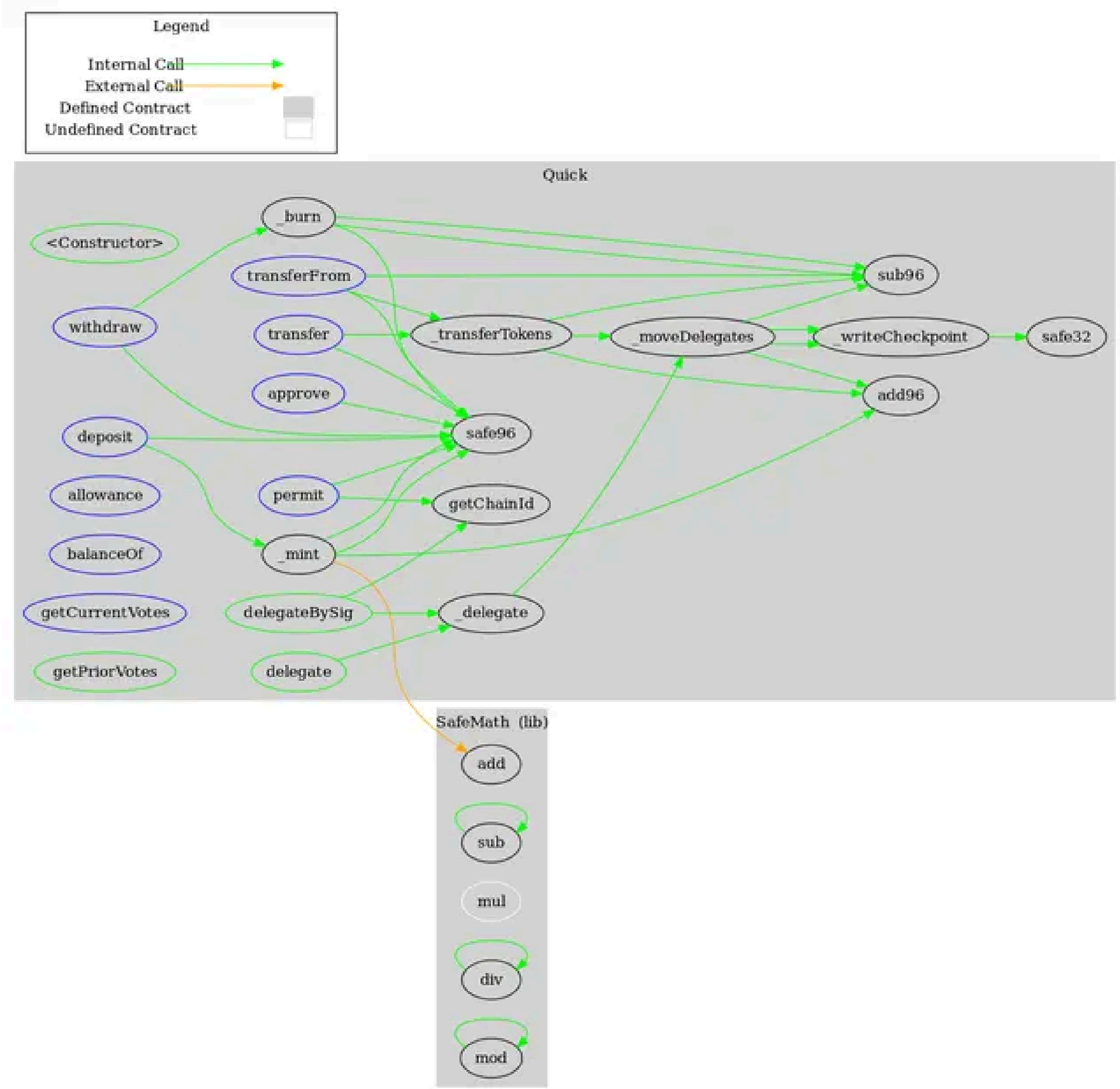
The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest 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**
SafeMath	Library			
	add	Internal		
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Quick	Implementation			
		Public	✓	-
	deposit	External	✓	-
	withdraw	External	✓	-
	allowance	External		-
	approve	External	✓	-
	permit	External	✓	-

	balanceOf	External		-
	transfer	External	✓	-
	transferFrom	External	✓	-
	delegate	Public	✓	-
	delegateBySig	Public	✓	-
	getCurrentVotes	External		-
	getPriorVotes	Public		-
	_delegate	Internal	✓	
	_transferTokens	Internal	✓	
	_moveDelegates	Internal	✓	
	_writeCheckpoint	Internal	✓	
	safe32	Internal		
	safe96	Internal		
	add96	Internal		
	sub96	Internal		
	getChainId	Internal		
	_mint	Internal	✓	
	_burn	Internal	✓	

Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Quick
<Constructor>
withdraw
deposit
allowance
balanceOf
getCurrentVotes
getPriorVotes
_burn
transferFrom
transfer
approve
permit
_mint
delegateBySig
delegate
_transferTokens
safe96
getChainId
_delegate
_moveDelegates
sub96
_writeCheckpoint
add96
safe32
SafeMath (lib)
add
sub
mul
div
mod

Summary

Quickswap contract implements a governance mechanism. This audit investigates security issues, business logic concerns and potential improvements.

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