Cyberscope

Audit Report

Tradecurve

April 2023

Network	ETH
Address	0x923607a06aa8f1c525a5bb2d921cf85e625be776
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Review 2
Audit Updates 2
Source Files 2
Findings Breakdown 3
Analysis 4
Diagnostics 5
IDI - Immutable Declaration Improvement 6
Description 6
Recommendation 6
L03 - Redundant Statements 7
Description 7
Recommendation 8
L09 - Dead Code Elimination 9
Description 9
Recommendation 9
Functions Analysis 10
Inheritance Graph 13
Flow Graph 14
Summary 15
Disclaimer 16
About Cyberscope 17

Review

Contract Name	Tradecurve
Compiler Version	v0.8.19+commit.7dd6d404
Optimization	200 runs
Explorer	https://etherscan.io/address/0x923607a06aa8f1c525a5bb2d921cf85e625be776
Address	0x923607a06aa8f1c525a5bb2d921cf85e625be776
Network	ETH
Symbol	TCRV
Decimals	18
Total Supply	1,800,000,000

Audit Updates

Initial Audit	07 Apr 2023
Corrected Phase 2	29 Apr 2023

Source Files

Filename	SHA256
Tradecurve.sol	b1677d16425651ad9ec7d94f34f32f3aed47cb0540ae1a41014b0ebe5f8ebd47

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	3

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

Analysis

Critical Medium Minor / Informative Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	IDI	Immutable Declaration Improvement	Unresolved
●	L03	Redundant Statements	Unresolved
●	L09	Dead Code Elimination	Unresolved

IDI - Immutable Declaration Improvement

Criticality	Minor / Informative
Location	Tradecurve.sol#L238,239
Status	Unresolved

Description

The contract uses variables that initialize them only in the constructor. The other functions are not mutating the variables. These variables are not defined as `immutable`.

```
_name
_symbol
```

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.

L03 - Redundant Statements

Criticality	Minor / Informative
Location	Tradecurve.sol#L217
Status	Unresolved

Description

Redundant statements are statements that are unnecessary or have no effect on the contract's behavior. These can include declarations of variables or functions that are not used or assignments to variables that are never used.

As a result, it can make the contract's code harder to read and maintain, and can also increase the contract's size and gas consumption, potentially making it more expensive to deploy and execute.

```
contract Tradecurve is Context, IERC20, IERC20Metadata, Ownable {
    mapping (address => uint256) private _balances;

    mapping (address => mapping (address => uint256)) private _allowances;

    uint256 private _totalSupply;
...
     * - when `to` is zero, `amount` of ``from``'s tokens will be burned.
     * - `from` and `to` are never both zero.
     *
     * To learn more about hooks, head to
xref:ROOT:extending-contracts.adoc#using-hooks[Using Hooks].
     */
    function _beforeTokenTransfer(address from, address to, uint256
amount) internal virtual { }
}
```

Recommendation

To avoid redundant statements, it's important to carefully review the contract's code and remove any statements that are unnecessary or not used. This can help to improve the clarity and efficiency of the contract's code.

By removing unnecessary or redundant statements from the contract's code, the clarity and efficiency of the contract will be improved. Additionally, the size and gas consumption will be reduced.

L09 - Dead Code Elimination

Criticality	Minor / Informative
Location	Tradecurve.sol#L439,454,466
Status	Unresolved

Description

In Solidity, dead code is code that is written in the contract, but is never executed or reached during normal contract execution. Dead code can occur for a variety of reasons, such as:

- Conditional statements that are always false.
- Functions that are never called.
- Unreachable code (e.g., code that follows a return statement).

Dead code can make a contract more difficult to understand and maintain, and can also increase the size of the contract and the cost of deploying and interacting with it.

```
function mint(uint256 amount) private onlyOwner {
        _mint(msg.sender, amount);
    }

...
```

Recommendation

To avoid creating dead code, it's important to carefully consider the logic and flow of the contract and to remove any code that is not needed or that is never executed. This can help improve the clarity and efficiency of the contract.

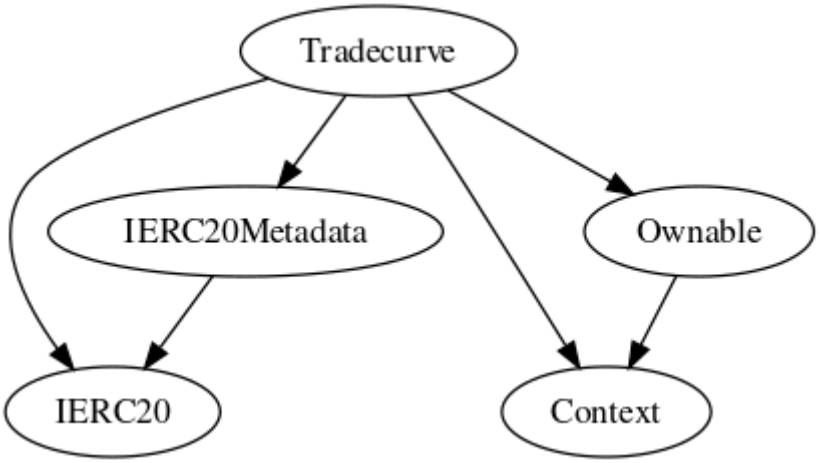
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		

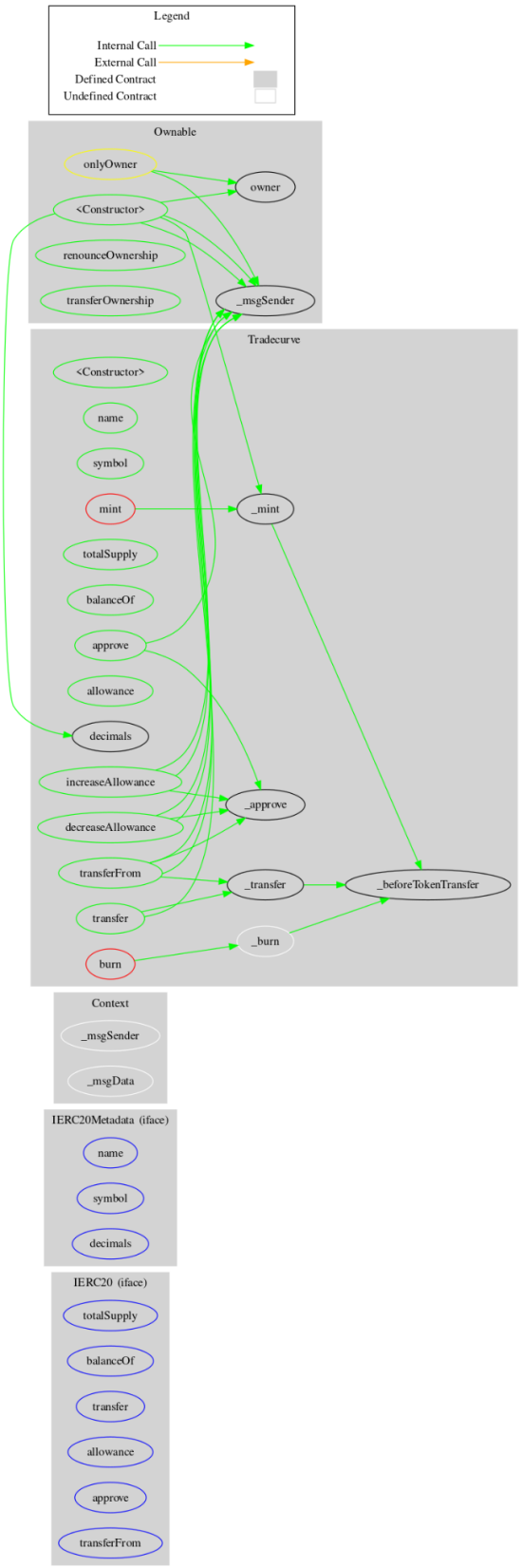
		Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
Tradecurve	Implementation	Context, IERC20, IERC20Metadata, Ownable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	mint	Private	✓	onlyOwner
	_burn	Internal	✓	

	burn	Private	✓	onlyOwner
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Ownable
onlyOwner
owner
<Constructor>
renounceOwnership
transferOwnership
_msgSender
Tradecurve
<Constructor>
name
symbol
mint
_mint
totalSupply
balanceOf
approve
allowance
decimals
increaseAllowance
_approve
decreaseAllowance
transferFrom
_transfer
_beforeTokenTransfer
transfer
_burn
burn
Context
_msgSender
_msgData
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom

Summary

Tradecurve contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. Tradecurve is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler errors 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.

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