Cyberscope

Audit Report

Minu

February 2023

Type	BEP20
Network	BSC
Address	0x0754088499311a3FC2A9D2B759Dab2b6c0dB4A15
Audited by	© cyberscope

Table of Contents

Review

Contract Name	MinuBones
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Explorer	https://bscscan.com/address/0x0754088499311a3fc2a9d2b759dab2b6c0db4a15
Address	0x0754088499311a3fc2a9d2b759dab2b6c0db4a15
Network	BSC

Audit Updates

Initial Audit	13 Feb 2023

Source Files

Filename	SHA256
contracts/MinuBones.sol	c97a5c9ce8cbd743b49e1161769869185a0945cc0d1e61ab6dbb9431d52d5622

Introduction

The Minu ecosystem consists of two smart contracts, an ERC20 Token contract, and a financial/staking contract. This audit report focuses on the MinuBones contract. To find out more about the MinuToken contract follow the link below: https://github.com/cyberscope-io/audits/tree/main/11-minu/Token.pdf

The primary objective of the game is to outpace other players by frequently recruiting additional miners, thereby accelerating BNB earnings. Utilizing daily BNB earnings for hiring miners will result in a threefold increase in the miner count within 30 days or less. Additionally, there is a 4% developer fee for each transaction, and the contract allows trades to specific addresses depending on its `tradingState` variable.

- When `tradingState` is equal to 0, then only the contract's owner can trade.
- When `tradingState` is equal to 1, then only presale users or the contract's owner can trade.
- When `tradingState` is equal to 2, then all Minu token holders can trade.

Lastly, the MinuBones contract has a direct, one-level referral system that rewards the referrer with 12% referral rewards, when invited users deposit and withdraw their tokens.

Roles

Owner

- `function setTradingState(uint8 _tradingState)`
- `function setToken(address _token)`
- `function seedMarket()`

User

- `function hatchBones(address ref)`
- `function sellBones()`
- `function beanRewards(address adr)`
- `function buyBones(address ref)`
- `function calculateBoneSell(uint256 bones)`
- `function calculateBoneBuy(uint256 eth, uint256 contractBalance)`
- `function calculateBoneBuySimple(uint256 eth)`
- `function getBalance()`
- `function getMyMiners(address adr)`
- `function getMyBones(address adr)`
- `function getBonesSinceLastHatch(address adr)`

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	L02	State Variables could be Declared Constant	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L16	Validate Variable Setters	Unresolved

L02 - State Variables could be Declared Constant

Criticality	Minor / Informative
Location	MinuBones.sol#L156,157,158,159,161
Status	Unresolved

Description

State variables can be declared as constant using the constant keyword. This means that the value of the state variable cannot be changed after it has been set. Additionally, the constant variables decrease gas consumption of the corresponding transaction.

```
uint256 private BONES_TO_HATCH_1MINERS = 1080000
uint256 private PSN = 10000
uint256 private PSNH = 5000
uint256 private devFeeVal = 4
address payable private recAdd = payable(0x4f5E3C8b92dB6e10ee49b030E98473d654051AAd)
```

Recommendation

Constant state variables can be useful when the contract wants to ensure that the value of a state variable cannot be changed by any function in the contract. This can be useful for storing values that are important to the contract's behavior, such as the contract's address or the maximum number of times a certain function can be called. The team is advised to add the constant keyword to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	MinuBones.sol#L156,157,158,185,190
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.

```
uint256 private BONES_TO_HATCH_1MINERS = 1080000
uint256 private PSN = 10000
uint256 private PSNH = 5000
uint8 _tradingState
address _token
```

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/v0.8.17/style-guide.html#naming-convention.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	MinuBones.sol#L118
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.

```
_owner = msgSender
```

Recommendation

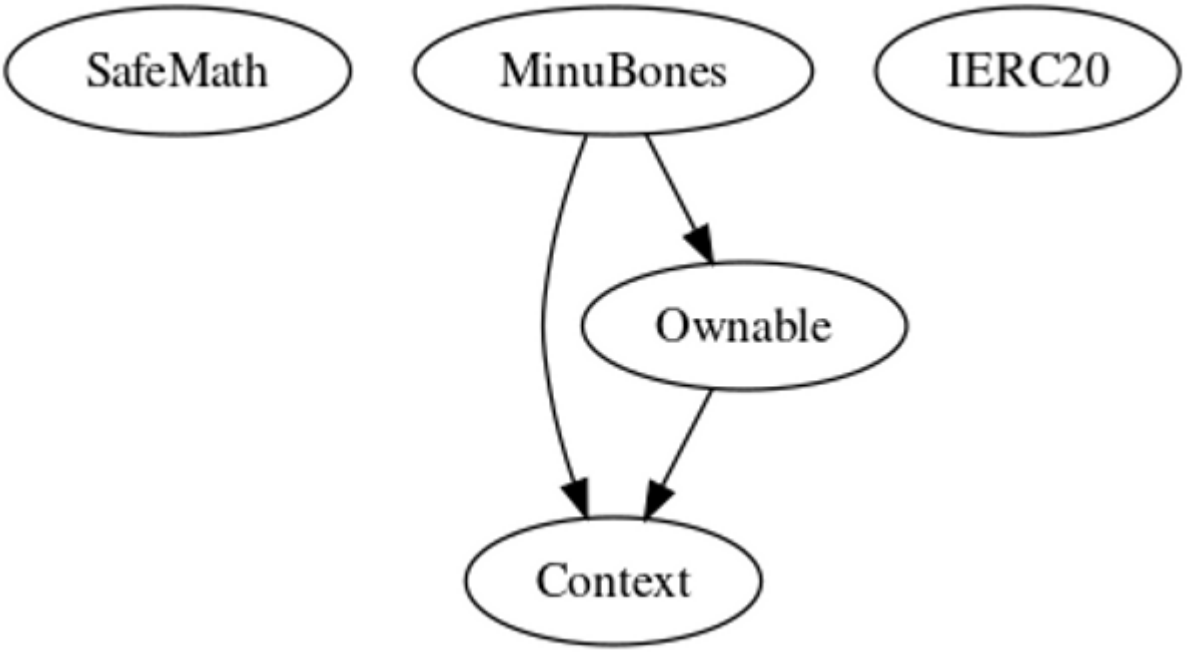
By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

Functions Analysis

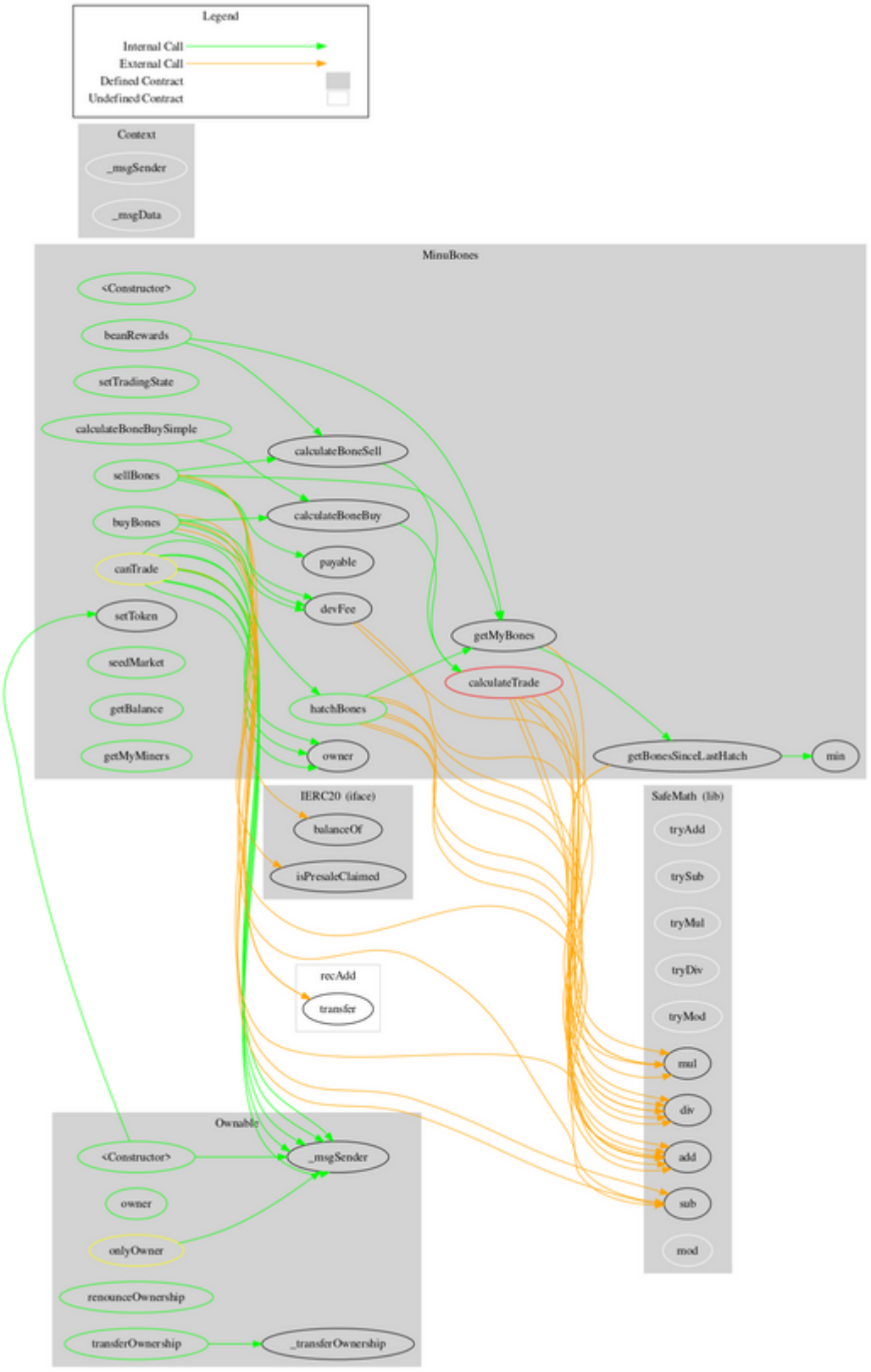
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner

	_transferOwnership	Internal	✓	
IERC20	Interface			
	balanceOf	External		-
	isPresaleClaimed	External		-
MineBones	Implementation	Context, Ownable		
		Public	✓	-
	setTradingState	Public	✓	onlyOwner
	setToken	Public	✓	onlyOwner
	hatchBones	Public	✓	canTrade
	sellBones	Public	✓	canTrade
	beanRewards	Public		-
	buyBones	Public	Payable	canTrade
	calculateTrade	Private		
	calculateBoneSell	Public		-
	calculateBoneBuy	Public		-
	calculateBoneBuySimple	Public		-
	devFee	Private		
	seedMarket	Public	Payable	onlyOwner
	getBalance	Public		-
	getMyMiners	Public		-
	getMyBones	Public		-
	getBonesSinceLastHatch	Public		-
	min	Private		

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Context
_msgSender
_msgData
MinuBones
<Constructor>
beanRewards
setTradingState
calculateBoneBuySimple
calculateBoneSell
sellBones
calculateBoneBuy
buyBones
payable
canTrade
devFee
setToken
getMyBones
seedMarket
calculateTrade
getBalance
hatchBones
getMyMiners
owner
getBonesSinceLastHatch
min
IERC20 (iface)
balanceOf
isPresaleClaimed
SafeMath (lib)
tryAdd
trySub
tryMul
tryDiv
tryMod
recAdd
transfer
mul
div
add
sub
mod
Ownable
<Constructor>
_msgSender
owner
onlyOwner
renounceOwnership
transferOwnership
_transferOwnership

Summary

Minu contract implements a financial and staking 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