



Cyberscope

Audit Report

Harmony Nodes

March 2022

Type ERC20

SHA256 3e3054b54a644324e36cd7c9e27e75471dea17d35a676b644820c4679fe1d396

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Audit Updates	3
Harmony Nodes Workflow	4
Contract Analysis	5
ST - Stop Transactions	6
Description	6
Recommendation	6
Contract Diagnostics	7
AP - Admin Privileges	8
Description	8
Recommendation	8
RSC - Reward System Concern	9
Description	9
Recommendation	10
MNO - Mint Node Overcharge	11
Description	11
Recommendation	11
FNM - Function Naming Misused	12
Description	12
Recommendation	12
CO - Code Optimization	13
Description	13
Recommendation	13
CR - Code Repetition	14
Description	14

Recommendation	14
MC - Missing Check	15
Description	15
Recommendation	16
L01 - Public Function could be Declared External	17
Description	17
Recommendation	17
L04 - Conformance to Solidity Naming Conventions	18
Description	18
Recommendation	18
L07 - Missing Events Arithmetic	19
Description	19
Recommendation	19
L08 - Tautology or Contradiction	20
Description	20
Recommendation	20
Contract Functions	21
Contract Flow	23
Domain Info	24
Summary	25
Disclaimer	26
About Cyberscope	27

Contract Review

SHA256	3e3054b54a644324e36cd7c9e27e75471dea17d35a676b644820c4679fe1d396
Source	HarmonyNodes.sol
Domain	harmonynodes.com

Audit Updates

Initial Audit	14th March 2022
Corrected	16th March 2022
Corrected	27th March 2022

Harmony Nodes Workflow

Harmony nodes tokens implement a reward mechanism. Users have the ability to create nodes. The user pays in HONE tokens in order to get a node. The nodes have a variation of cost and interest multipliers. The interest multiplier is increased logarithmically to the node's cost. The nodes cost / interest multiplier ratio is the following:

No	Type	Node Cost	Multiplier
0	Nano	10	1
2	Pico	20	2
3	Mega	50	5
4	Giga	100	10

Each address can update the owned node without limit. For instance, an address could have one Nano and two Mega nodes.

Contract Analysis

● Critical ● Medium ● Minor ● Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	minor
Location	contract.sol#L499,504

Description

The contract owner has the authority to stop transactions for all users. The owner may take advantage of it by setting the `limit` to zero.

```
require(amount <= limit, 'This transfer exceeds the allowed limit!');
```

Recommendation

The contract could embody a check for not allowing setting the limit less than a reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	AP	Admin Privileges
●	RSC	Reward System Concern
●	MNO	Mint Node Overcharge
●	FNM	Function Naming Misused
●	CO	Code Optimization
●	CR	Code Repetition
●	MC	Missing Check
●	L01	Public Function could be Declared External
●	L04	Conformance to Solidity Naming Conventions
●	L07	Missing Events Arithmetic
●	L08	Tautology or Contradiction

AP - Admin Privileges

Criticality	medium
Location	contract.sol#L548,658

Description

- The dev addresses will initially get an interest multiplier for 5 Giga nodes without paying.
- The contract owner has the ability to reserve nodes for any address without paying the corresponding fees.

```
for(i=0; i < _devs.length; i++){  
    HONENodesAddresses.push(_devs[i]);  
    Account memory account = Account(true, 0, 0, 0, 5, 0);  
    accounts[_devs[i]] = account;  
    totalNodes += 5;  
}
```

Recommendation

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one. The owner could also renounce the contract ownership for a period of time or pass the access to the zero address.

RSC - Reward System Concern

Criticality	medium
Location	contract.sol#L630

Description

The reward system contains a mint/burn mechanism that may inflate or deflate the token balance and the node holders. For instance:

Example 1

HONEManager balance: 10,000 HONE

1. A user buy a Giga node by paying 500,000 HONE.
2. The HONEManager will be 510,000.
3. The owner triggers the *manageRewards()* function.

The following will happen:

poolAmount = 500000

runwayTime = 50000

newTotalTokens = 3650

amountToBurn = 463500

poolAmount = 496350

That means that the corresponding user's tokens will be burned.

Example 2

On the other hand, the contract owner may trigger the *manageRewards()* in an early state, mint new tokens and use the *burnHONE()* function in order to transfer these tokens to an address.

```
function manageRewards() public {
    require(msg.sender == owner, 'Only the owner can call this.');
```



```
    uint poolAmount = HONEAddress.balanceOf(address(this)) / 10 ** 18;
    uint runwayTime = poolAmount / ((totalNodes * HONEInterestRatePercent *
nodeTierCoefficient[3]) / 100);
    if(runwayTime > 730){
        uint newTotalTokens = (365 * HONEInterestRatePercent * totalNodes *
nodeTierCoefficient[3]) / 100;
        uint amountToBurn = poolAmount - newTotalTokens;
        HONEAddress.burn(amountToBurn * 10 ** 18);
    }
    else if(runwayTime < 360){
        uint newTotalTokens = (365 * HONEInterestRatePercent * totalNodes *
nodeTierCoefficient[3]) / 100;
        uint amountToMint = newTotalTokens - poolAmount;
        HONEAddress.mint(amountToMint * 10 ** 18);
    }
}
```

Recommendation

A more clear approach for minting/burning tokens could be introduced. Additionally, the namings could be changed to something more related to the business logic. For instance the *runwayInDays* does not contain any data that is related to the days.

MNO - Mint Node Overcharge

Criticality	medium
Location	contract.sol#L556

Description

The `mintHarmonyNode()` provides a node according to a charge. If the issuer provides more tokens than the specification, they are all transferred to the contract. The `mintHarmonyNode()` does not provide a way to track these amounts. Thus, the user will essentially pay more than the required without any reward.

```
if(_nodeType == 0){  
    require(_HONEAmount >= 10 * 10 ** 18, 'You must provide at least 10 HONE');  
  
    account.nanoCount++;  
}  
...  
HONEAddress.transferFrom(_address, address(this), _HONEAmount);
```

Recommendation

The `mintHarmonyNode()` should transfer only the amount of tokens that is required for the specific node type.

FNM - Function Naming Misused

Criticality	minor
Location	contract.sol#L652

Description

The `burnHONE()` naming provides the perception that the provided amount of HOME will be burned. In contrast, the function sent from the contract to the target address the provided amount of HONE.

```
function burnHONE(address _zero, uint amount) public {  
    require(msg.sender == owner, 'Only the owner can call this.');
```

```
    HONEAddress.transfer(_zero, amount);  
}
```

Recommendation

The naming of the function should be changed to something more related to the function's business logic.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L556,597

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The contract checks if the target address is the sender.

```
function mintHarmonyNode(address _address, uint _HONEAmount, uint _nodeType)
public {
    require(msg.sender == _address, 'Only user can create a node.');
```



```
function claimYield(address _to) public {
    require(msg.sender == _to, 'Only user can withdraw its own funds.');
```

Recommendation

Since the only allowed address is the sender, then the argument could be eliminated and use the `msg.sender` directly.

CR - Code Repetition

Criticality	minor
Location	contract.sol#L663

Description

There are code segments that are repetitive in the contract. Those segments increase the code size of the contract unnecessarily.

For instance, the following code segment is used in `mintHarmonyNode()` and `awardNode()` as well.

```
Account memory account;

if(accounts[_address].exists){
    account = accounts[_address];
}
else{
    account = Account(true, 0, 0, 0, 0, 0);
    HONENodesAddresses.push(_address);
}
```

Recommendation

Create an internal function that contains the code segment and remove it from all the sections.

MC - Missing Check

Criticality	critical
Location	contract.sol#L597,658

Description

The contract is processing variables that have not properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues.

The transfer is triggered without checking if it has completed successfully. In case of failure, the issuer will lose the accumulated interests without receiving them.

```
function claimYield(address _to) public {
    require(msg.sender == _to, 'Only user can withdraw its own funds.');
```

require(accounts[_to].interestAccumulated > 0, 'Interest accumulated must be greater than zero.');

```
    uint amount = accounts[_to].interestAccumulated;
    accounts[_to].interestAccumulated = 0;

    HONEAddress.transfer(_to, amount);
}
```

The *_nodeType* should be a number less or equal to 4, otherwise the totalNodes will be increased without the corresponding counter update.

```
function awardNode(address _address, uint _nodeType) public {
    require(msg.sender == owner, 'You must be the owner to run this.');
```

The *HONEInterestRatePercent* should not be allowed to be zero because it is used as a divider in the calculations.

```
function changeDailyYield(uint _newRate) public {
    require(msg.sender == owner, 'You must be the owner to run this.');
```

HONEInterestRatePercent = _newRate;

```
}
```

///

```
uint runwayInDays = poolAmount/((totalNodes * HONEInterestRatePercent *
```



```
nodeMultipliers[4]) / 100);
```

Recommendation

The contract should properly check the variables according to the required specifications

L01 - Public Function could be Declared External

Criticality

minor

Location

contract.sol#L181,189,206,213,220,240,251,297,316,488 and 10 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
awardNode  
burnHONE  
changeDailyYield  
manageRewards  
payYield  
claimYield  
mintHarmonyNode  
burn  
mint  
...
```

Recommendation

Use the external attribute for functions never called from the contract

L04 - Conformance to Solidity Naming Conventions

Criticality

minor

Location

contract.sol#L488,493,508,513,479,556,597,647,652,658 and 6 more

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow `_` at the beginning of the `mixed_case` match for private variables and unused parameters.

```
HONEInterestRatePercent
HONEAddress
HONENodesAddresses
_nodeType
_address
_zero
_newRate
_to
_HONEAmount
...
```

Recommendation

Follow the Solidity naming convention.

<https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions>

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L493

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
limit = _limit
```

Recommendation

Emit an event for critical parameter changes.

L08 - Tautology or Contradiction

Criticality

minor

Location

contract.sol#L556

Description

Detects expressions that are tautologies or contradictions. For instance, an uint variable will always be greater than or equal to zero.

```
require(bool,string)(_nodeType >= 0 && _nodeType <= 3,Invalid node tier)
```

Recommendation

Fix the incorrect comparison by changing the value type or the comparison.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
ERC20	Implementation	Context, IERC20, IERC20Meta data		
	<Constructor>	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	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
HONE	Implementation	ERC20		
	<Constructor>	Public	✓	ERC20
	setManagerContract	Public	✓	-
	setTransferLimit	Public	✓	-
	transferFrom	Public	✓	-
	transfer	Public	✓	-
	burn	Public	✓	-
HONEManager	Implementation			
	<Constructor>	Public	✓	-
	mintHarmonyNode	Public	✓	-
	claimYield	Public	✓	-
	payYield	Public	✓	-
	manageRewards	Public	✓	-
	changeDailyYield	Public	✓	-
	burnHONE	Public	✓	-
	awardNode	Public	✓	-

Contract Flow



Domain Info

Domain Name	harmonynodes.com
Registry Domain ID	2678732538_DOMAIN_COM-VRSN
Creation Date	2022-03-02T07:29:09Z
Updated Date	2022-03-04T03:12:34Z
Registry Expiry Date	2023-03-02T07:29:09Z
Registrar WHOIS Server	whois.godaddy.com
Registrar URL	http://www.godaddy.com
Registrar	GoDaddy.com, LLC
Registrar IANA ID	146

The domain has been created 22 days before the creation of the audit. It will expire in 12 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

The Harmony Nodes is a contract that contains a standard token functionality enriched with a reward mechanism based on nodes acquisition. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

The Nodes acquisition mechanism contains some flows that may affect the expected behaviour. We mention some concerns regarding the contract owner privileges.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provides all the essential tools to assist users draw their own conclusions.



The Cyberscope team

<https://www.cyberscope.io>