

Coinhound Smart Contract Audit Report



22 Jan 2023



TABLE OF CONTENTS

Audited Details

- Audited Project
- Blockchain
- Addresses
- Project Website
- Codebase

Summary

- Contract Summary
- Audit Findings Summary
- Vulnerabilities Summary

Conclusion

Audit Results

Smart Contract Analysis

- Detected Vulnerabilities

Disclaimer

About Us



AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain	
Coinhound	CND	BSC	

Addresses

Contract address	0x4a7DbA30250aBE3CAbe3d3809CF843aa0394623F
Contract deployer address	0xf4D0bE5E827f0cB2EF7c1165560990B1e63A9251

Project Website

https://coinhound.io/

Codebase

https://bscscan.com/address/0x4a7DbA30250aBE3CAbe3d3809CF843aa0394623F#code



SUMMARY

Coinhound is on a mission, to make the decentralized space fun, safer, and smarter for everyone. Coinhound is the first GameFi dApp (Decentralized application) that users can use to scan and track important information all across the blockchain network. We are community driven, one of a kind, and committed to delivering a game changing product. Unlocked tokens are now locked in Mudra locker, check our Telegram for TXN.

Contract Summary

Documentation Quality

Coinhound provides a document with an excellent standard of solidity base code.

• The technical description is clearly structured and has some low-risk issues.

Code Quality

The Overall quality of the basecode is GOOD

• Standard solidity basecode and rules are already followed with the Coinhound project.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

SWC-101 | Arithmetic operation Issues discovered on lines 445, 455, 463, 482, 484, 496, 497, 511, 513, 612, 613, 633, 634, 636, 637, 671, 676, 678, 684, 689, 691, 752, 753, 754, 757, 762, 762, 781, 782, 797, 804, 805, 820, 848, 875, 876, 879, 880, 810, 811, 838, and 839.



CONCLUSION

We have audited the Coinhound project released on January 2023 to discover issues and identify potential security vulnerabilities in Coinhound Project. This process finds bugs, technical issues, and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with a low risk issue on the contract project.

The most common issue found in writing code on contracts that do not pose a big risk, writing on contracts is close to the standard of writing contracts in general. Some of the common issues that were found were integer overflow and underflow. We recommend functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously



AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	PASS
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	PASS
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Check-Effect Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	PASS
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	PASS
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Caller	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS



Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	PASS
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	
Signature Unique Id	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS



SMART CONTRACT ANALYSIS

Started	Sat Jan 21 2023 02:23:08 GMT+0000 (Coordinated Universal Time)		
Finished	Sun Jan 22 2023 03:33:18 GMT+0000 (Coordinated Universal Time)		
Mode	Standard		
Main Source File	Cnd.Sol		

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged



SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged





SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged



LINE 445

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
444 unchecked {
445 __approve(sender, __msgSender(), currentAllowance - amount);
446 }
447 }
448
```



LINE 455

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
454 function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
455 _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
456 return true;
457 }
458
```



LINE 463

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
462 unchecked {
463 _approve(_msgSender(), spender, currentAllowance - subtractedValue);
464 }
465 
466 return true;
```



LINE 482

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
481 unchecked {
482 _balances[sender] = senderBalance - amount;
483 }
484 _balances[recipient] += amount;
485
```



LINE 484

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

Locations

483 }
483 }
484 _balances[recipient] += amount;
485
486 emit Transfer(sender, recipient, amount);
487



LINE 496

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

Locations

495
496 _totalSupply += amount;
497 _balances[account] += amount;
498 emit Transfer(address(0), account, amount);
499



LINE 497

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

Locations

496 _totalSupply += amount; 497 _balances[account] += amount; 498 emit Transfer(address(0), account, amount); 499 500 _afterTokenTransfer(address(0), account, amount);



LINE 511

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
510 unchecked {
511 _balances[account] = accountBalance - amount;
512 }
513 _totalSupply -= amount;
514
```



LINE 513

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
512 }
513 _totalSupply -= amount;
514
515 emit Transfer(account, address(0), amount);
516
```



LINE 612

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
611
612 _totalFeesOnBuy = liquidityFeeOnBuy + marketingFeeOnBuy;
613 _totalFeesOnSell = liquidityFeeOnSell + marketingFeeOnSell;
614
615 walletToWalletTransferFee = 0;
```



LINE 613

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
612 _totalFeesOnBuy = liquidityFeeOnBuy + marketingFeeOnBuy;
613 _totalFeesOnSell = liquidityFeeOnSell + marketingFeeOnSell;
614
615 walletToWalletTransferFee = 0;
616
```



LINE 633

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
632
633 _mint(owner(), 1e9 * (10 ** decimals()));
634 swapTokensAtAmount = totalSupply() / 5_000;
635
636 maxTransactionAmountBuy = totalSupply() * 20 / 1000;
```



LINE 634

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
633 __mint(owner(), le9 * (10 ** decimals()));
634 swapTokensAtAmount = totalSupply() / 5_000;
635
636 maxTransactionAmountBuy = totalSupply() * 20 / 1000;
637 maxTransactionAmountSell = totalSupply() * 20 / 1000;
```



LINE 636

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
635
636 maxTransactionAmountBuy = totalSupply() * 20 / 1000;
637 maxTransactionAmountSell = totalSupply() * 20 / 1000;
638
639 swapEnabled = true;
```



LINE 637

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
636 maxTransactionAmountBuy = totalSupply() * 20 / 1000;
637 maxTransactionAmountSell = totalSupply() * 20 / 1000;
638
639 swapEnabled = true;
640 }
```



LINE 671

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
670 function updateBuyFees(uint256 _liquidityFeeOnBuy, uint256 _marketingFeeOnBuy)
external onlyOwner {
671   uint256 oldFee = _totalFeesOnBuy + _totalFeesOnSell;
672
673   liquidityFeeOnBuy = _liquidityFeeOnBuy;
674   marketingFeeOnBuy = _marketingFeeOnBuy;
```



LINE 676

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
675
676 _totalFeesOnBuy = liquidityFeeOnBuy + marketingFeeOnBuy;
677
678 require(_totalFeesOnBuy + _totalFeesOnSell <= oldFee, "Total Fees cannot exceed the
old fee");
679</pre>
```



LINE 678

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
677
678 require(_totalFeesOnBuy + _totalFeesOnSell <= oldFee, "Total Fees cannot exceed the
old fee");
679
680 emit UpdateBuyFees(liquidityFeeOnBuy, marketingFeeOnBuy);
681 }</pre>
```



LINE 684

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
683 function updateSellFees(uint256 _liquidityFeeOnSell, uint256 _marketingFeeOnSell)
external onlyOwner {
684 uint256 oldFee = _totalFeesOnBuy + _totalFeesOnSell;
685
686 liquidityFeeOnSell = _liquidityFeeOnSell;
687 marketingFeeOnSell = _marketingFeeOnSell;
```



LINE 689

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
688
689 _totalFeesOnSell = liquidityFeeOnSell + marketingFeeOnSell;
690
691 require(_totalFeesOnBuy + _totalFeesOnSell <= oldFee, "Total Fees cannot exceed the
old fee");
692</pre>
```



LINE 691

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
690
691 require(_totalFeesOnBuy + _totalFeesOnSell <= oldFee, "Total Fees cannot exceed the
old fee");
692
693 emit UpdateSellFees(liquidityFeeOnSell, marketingFeeOnSell);
694 }</pre>
```



SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 747

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
746 to == uniswapV2Pair &&
747 _totalFeesOnBuy + _totalFeesOnSell > 0 &&
748 swapEnabled
749 ) {
750 swapping = true;
```



LINE 752

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
751
752 uint256 totalFee = _totalFeesOnBuy + _totalFeesOnSell;
753 uint256 liquidityShare = liquidityFeeOnBuy + liquidityFeeOnSell;
754 uint256 marketingShare = marketingFeeOnBuy + marketingFeeOnSell;
755
```



LINE 753

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
752 uint256 totalFee = _totalFeesOnBuy + _totalFeesOnSell;
753 uint256 liquidityShare = liquidityFeeOnBuy + liquidityFeeOnSell;
754 uint256 marketingShare = marketingFeeOnBuy + marketingFeeOnSell;
755
756 if (liquidityShare > 0) {
```



LINE 754

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
753 uint256 liquidityShare = liquidityFeeOnBuy + liquidityFeeOnSell;
754 uint256 marketingShare = marketingFeeOnBuy + marketingFeeOnSell;
755
756 if (liquidityShare > 0) {
757 uint256 liquidityTokens = contractTokenBalance * liquidityShare / totalFee;
```



LINE 757

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
756 if (liquidityShare > 0) {
757 uint256 liquidityTokens = contractTokenBalance * liquidityShare / totalFee;
758 swapAndLiquify(liquidityTokens);
759 }
760
```



LINE 762

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
761 if (marketingShare > 0) {
762 uint256 marketingTokens = contractTokenBalance * marketingShare / totalFee;
763 swapAndSendMarketing(marketingTokens);
764 }
765
```



LINE 762

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
761 if (marketingShare > 0) {
762 uint256 marketingTokens = contractTokenBalance * marketingShare / totalFee;
763 swapAndSendMarketing(marketingTokens);
764 }
765
```



LINE 781

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
780 if (_totalFees > 0) {
781 uint256 fees = (amount * _totalFees) / 100;
782 amount = amount - fees;
783 super._transfer(from, address(this), fees);
784 }
```



LINE 782

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
781 uint256 fees = (amount * _totalFees) / 100;
782 amount = amount - fees;
783 super._transfer(from, address(this), fees);
784 }
785
```



LINE 797

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

Locations

796 function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner{ 797 require(newAmount > totalSupply() / 1_000_000, "SwapTokensAtAmount must be greater than 0.0001% of total supply"); 798 swapTokensAtAmount = newAmount; 799 800 emit SwapTokensAtAmountUpdated(swapTokensAtAmount);



LINE 804

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
803 function swapAndLiquify(uint256 tokens) private {
804 uint256 half = tokens / 2;
805 uint256 otherHalf = tokens - half;
806
807 uint256 initialBalance = address(this).balance;
```



LINE 805

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
804 uint256 half = tokens / 2;
805 uint256 otherHalf = tokens - half;
806
807 uint256 initialBalance = address(this).balance;
808
```



LINE 820

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

Locations

819
820 uint256 newBalance = address(this).balance - initialBalance;
821
822 uniswapV2Router.addLiquidityETH{value: newBalance}(
823 address(this),



LINE 848

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

Locations

847
848 uint256 newBalance = address(this).balance - initialBalance;
849
850 payable(marketingWallet).sendValue(newBalance);
851



LINE 875

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
874 require(
875 _maxTransactionAmountBuy >= (totalSupply() / (10 ** decimals())) / 1_000 &&
876 _maxTransactionAmountSell >= (totalSupply() / (10 ** decimals())) / 1_000,
877 "Max Transaction limis cannot be lower than 0.1% of total supply"
878 );
```



LINE 876

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
875 _maxTransactionAmountBuy >= (totalSupply() / (10 ** decimals())) / 1_000 &&
876 _maxTransactionAmountSell >= (totalSupply() / (10 ** decimals())) / 1_000,
877 "Max Transaction limis cannot be lower than 0.1% of total supply"
878 );
879 maxTransactionAmountBuy = _maxTransactionAmountBuy * (10 ** decimals());
```



LINE 879

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
878 );
879 maxTransactionAmountBuy = _maxTransactionAmountBuy * (10 ** decimals());
880 maxTransactionAmountSell = _maxTransactionAmountSell * (10 ** decimals());
881
882 emit MaxTransactionLimitAmountChanged(maxTransactionAmountBuy,
maxTransactionAmountSell);
```



LINE 880

Iow SEVERITY

This plugin produces issues to support false positive discovery within Mythril.

Source File

- cnd.sol

```
879 maxTransactionAmountBuy = _maxTransactionAmountBuy * (10 ** decimals());
880 maxTransactionAmountSell = _maxTransactionAmountSell * (10 ** decimals());
881
882 emit MaxTransactionLimitAmountChanged(maxTransactionAmountBuy,
maxTransactionAmountSell);
883 }
```



LINE 810

Iow SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- cnd.sol

```
809 address[] memory path = new address[](2);
810 path[0] = address(this);
811 path[1] = uniswapV2Router.WETH();
812
813 uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens()
```



LINE 811

Iow SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- cnd.sol

```
810 path[0] = address(this);
811 path[1] = uniswapV2Router.WETH();
812
813 uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
814 half,
```



LINE 838

Iow SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- cnd.sol

```
837 address[] memory path = new address[](2);
838 path[0] = address(this);
839 path[1] = uniswapV2Router.WETH();
840
841 uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens()
```



LINE 839

Iow SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- cnd.sol

```
838 path[0] = address(this);
839 path[1] = uniswapV2Router.WETH();
840
841 uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
842 tokenAmount,
```



DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. 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 Sysfixed's prior written consent in each instance.

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 considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Sysfixed to perform a security assessment. This report 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 is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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.

This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.



ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.