



FUTU

Smart Contract Audit Report

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
FUTU	FUTU	BSC

Addresses

Contract address	0x320953724bf0361fd0cDBf9eA9A23c7D4116FCb8
Contract deployer address	0xB5eA2A87c2AB144993ce452716e935865e2B0eED

Project Website

<http://futu.com/>

Codebase

<https://bscscan.com/address/0x320953724bf0361fd0cDBf9eA9A23c7D4116FCb8#contracts>

SUMMARY

One-stop crypto investment, social intelligence, state-of-the-art community-driven resource app with DeFi built tools for BNBCChain.

Contract Summary

Documentation Quality

This project has a standard of documentation.

- Technical description provided.

Code Quality

The quality of the code in this project is up to standard.

- The official Solidity style guide is followed.

Test Scope

Project test coverage is 100% (Via Codebase).

Audit Findings Summary

Issues Found

- SWC-101 | Arithmetic operation issues discovered on lines 555, 578, 611, 613, 634, 635, 660, 662, 711, 885, 913, 939, 942, 950, 960, 961, 962, 968, 969, 970, 977, 982, 1023, 1027, 1032, and 1037.
- SWC-103 | A floating pragma is set on lines 10, 235, 320, 350, 377, and 762. It changed from `^0.8.1` to `^0.8.0`.
- SWC-108 | State variable visibility is not set on lines 899 and 900. It is best practice to set the visibility of state variables explicitly to public or private.
- SWC-110 | Out of bounds array access issues discovered on lines 993 and 994.

CONCLUSION

We have audited the FUTU project which has released on January 2023 to discover issues and identify potential security vulnerabilities in FUTU Project. This process is used to find technical issues and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with low-risk issues.

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. The low-level issues found are a floating pragma is set on some lines, state variable visibility is not set on some lines, and out of bounds array access which the index access expression can cause an exception in case of use of an invalid array index value.

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.	ISSUE FOUND
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.	ISSUE FOUND
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.	ISSUE FOUND
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.	PASS
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.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	PASS
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	Wed Jan 18 2023 22:37:20 GMT+0000 (Coordinated Universal Time)
Finished	Thu Jan 19 2023 00:17:57 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	futu.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-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 555

low SEVERITY

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

Source File

- futu.sol

Locations

```
554 address owner = _msgSender();
555 _approve(owner, spender, allowance(owner, spender) + addedValue);
556 return true;
557 }
558
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 578

low SEVERITY

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

Source File

- futu.sol

Locations

```
577 unchecked {  
578   _approve(owner, spender, currentAllowance - subtractedValue);  
579 }  
580  
581 return true;
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 611

low SEVERITY

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

Source File

- futu.sol

Locations

```
610     unchecked {  
611         _balances[from] = fromBalance - amount;  
612     }  
613     _balances[to] += amount;  
614
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 613

low SEVERITY

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

Source File

- futu.sol

Locations

```
612     }  
613     _balances[to] += amount;  
614  
615     emit Transfer(from, to, amount);  
616
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 634

low SEVERITY

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

Source File

- futu.sol

Locations

```
633
634  _totalSupply += amount;
635  _balances[account] += amount;
636  emit Transfer(address(0), account, amount);
637
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 635

low SEVERITY

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

Source File

- futu.sol

Locations

```
634  _totalSupply += amount;  
635  _balances[account] += amount;  
636  emit Transfer(address(0), account, amount);  
637  
638  _afterTokenTransfer(address(0), account, amount);
```


SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 660

low SEVERITY

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

Source File

- futu.sol

Locations

```
659     unchecked {  
660         _balances[account] = accountBalance - amount;  
661     }  
662     _totalSupply -= amount;  
663
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 662

low SEVERITY

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

Source File

- futu.sol

Locations

```
661     }
662     _totalSupply -= amount;
663
664     emit Transfer(account, address(0), amount);
665
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 711

low SEVERITY

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

Source File

- futu.sol

Locations

```
710     unchecked {  
711         _approve(owner, spender, currentAllowance - amount);  
712     }  
713 }  
714 }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 885

low SEVERITY

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

Source File

- futu.sol

Locations

```
884
885  uint256 public swapThreshold = 50_000 * 10e18;
886  uint256 public maxBuy;
887
888  address public marketingWallet;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 913

low SEVERITY

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

Source File

- futu.sol

Locations

```
912 constructor() ERC20("FUTU", "FUTU") {
913   _mint(msg.sender, 100000 * 10 ** decimals());
914   excludedFromFees[msg.sender] = true;
915
916   IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 939

low SEVERITY

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

Source File

- futu.sol

Locations

```
938
939     else if(recipient == pair) fee = amount * totalSellTax / 100;
940     else if(sender == pair){
941         require(amount <= maxBuy, "You are exceeding maxBuy");
942         fee = amount * totalBuyTax / 100;
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 942

low SEVERITY

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

Source File

- futu.sol

Locations

```
941   require(amount <= maxBuy, "You are exceeding maxBuy");
942   fee = amount * totalBuyTax / 100;
943   }
944
945
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 950

low SEVERITY

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

Source File

- futu.sol

Locations

```
949
950  super._transfer(sender, recipient, amount - fee);
951  if(fee > 0) super._transfer(sender, address(this) ,fee);
952
953  }
```


SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 960

low SEVERITY

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

Source File

- futu.sol

Locations

```
959 // Split the contract balance into halves
960 uint256 denominator = totalSellTax * 2;
961 uint256 tokensToAddLiquidityWith = contractBalance * sellTaxes.liquidity /
denominator;
962 uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
963
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 961

low SEVERITY

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

Source File

- futu.sol

Locations

```
960 uint256 denominator = totalSellTax * 2;
961 uint256 tokensToAddLiquidityWith = contractBalance * sellTaxes.liquidity /
denominator;
962 uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
963
964 uint256 initialBalance = address(this).balance;
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 962

low SEVERITY

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

Source File

- futu.sol

Locations

```
961  uint256 tokensToAddLiquidityWith = contractBalance * sellTaxes.liquidity /  
denominator;  
962  uint256 toSwap = contractBalance - tokensToAddLiquidityWith;  
963  
964  uint256 initialBalance = address(this).balance;  
965
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 968

low SEVERITY

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

Source File

- futu.sol

Locations

```
967
968  uint256 deltaBalance = address(this).balance - initialBalance;
969  uint256 unitBalance= deltaBalance / (denominator - sellTaxes.liquidity);
970  uint256 bnbToAddLiquidityWith = unitBalance * sellTaxes.liquidity;
971
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 969

low SEVERITY

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

Source File

- futu.sol

Locations

```
968 uint256 deltaBalance = address(this).balance - initialBalance;
969 uint256 unitBalance= deltaBalance / (denominator - sellTaxes.liquidity);
970 uint256 bnbToAddLiquidityWith = unitBalance * sellTaxes.liquidity;
971
972 if(bnbToAddLiquidityWith > 0){
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 970

low SEVERITY

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

Source File

- futu.sol

Locations

```
969 uint256 unitBalance= deltaBalance / (denominator - sellTaxes.liquidity);
970 uint256 bnbToAddLiquidityWith = unitBalance * sellTaxes.liquidity;
971
972 if(bnbToAddLiquidityWith > 0){
973 // Add liquidity to pancake
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 977

low SEVERITY

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

Source File

- futu.sol

Locations

```
976
977     uint256 marketingAmt = unitBalance * 2 * sellTaxes.marketing;
978     if(marketingAmt > 0){
979         payable(marketingWallet).sendValue(marketingAmt);
980     }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 982

low SEVERITY

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

Source File

- futu.sol

Locations

```
981
982 uint256 devAmt = unitBalance * 2 * sellTaxes.dev;
983 if(devAmt > 0){
984 payable(devFunds).sendValue(devAmt);
985 }
```


SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1023

low SEVERITY

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

Source File

- futu.sol

Locations

```
1022 function setSwapThreshold(uint256 new_amount) external onlyOwner {
1023     swapThreshold = new_amount * 10**decimals();
1024 }
1025
1026 function setMaxBuy(uint256 amount) external onlyOwner{
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1027

low SEVERITY

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

Source File

- futu.sol

Locations

```
1026 function setMaxBuy(uint256 amount) external onlyOwner{
1027     maxBuy = amount * 10**decimals();
1028 }
1029
1030 function setBuyTaxes(uint256 _marketing, uint256 _dev, uint256 _liquidity)
external onlyOwner{
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1032

low SEVERITY

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

Source File

- futu.sol

Locations

```
1031 buyTaxes = Taxes(_marketing, _dev, _liquidity);
1032 totalBuyTax = _marketing + _dev + _liquidity;
1033 }
1034
1035 function setSellTaxes(uint256 _marketing, uint256 _dev, uint256 _liquidity)
external onlyOwner{
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1037

low SEVERITY

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

Source File

- futu.sol

Locations

```
1036     sellTaxes = Taxes(_marketing, _dev, _liquidity);
1037     totalSellTax = _marketing + _dev + _liquidity;
1038 }
1039
1040
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 10

low SEVERITY

The current pragma Solidity directive is ""^0.8.1"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- futu.sol

Locations

```
9
10  pragma solidity ^0.8.1;
11
12  /**
13   * @dev Collection of functions related to the address type
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 235

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- futu.sol

Locations

```
234
235  pragma solidity ^0.8.0;
236
237  /**
238  * @dev Interface of the ERC20 standard as defined in the EIP.
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 320

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- futu.sol

Locations

```
319
320  pragma solidity ^0.8.0;
321
322
323  /**
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 350

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- futu.sol

Locations

```
349
350 pragma solidity ^0.8.0;
351
352 /**
353  * @dev Provides information about the current execution context, including the
```


SWC-103 | A FLOATING PRAGMA IS SET.

LINE 377

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- futu.sol

Locations

```
376  
377  pragma solidity ^0.8.0;  
378  
379  
380
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 762

low SEVERITY

The current pragma Solidity directive is ""^0.8.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- futu.sol

Locations

```
761
762  pragma solidity ^0.8.0;
763
764
765  /**
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 899

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "totalBuyTax" is internal. Other possible visibility settings are public and private.

Source File

- futu.sol

Locations

```
898 Taxes public sellTaxes = Taxes(1,1,1);
899 uint256 totalBuyTax = 3;
900 uint256 totalSellTax = 3;
901
902 mapping (address => bool) public excludedFromFees;
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 900

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "totalSellTax" is internal. Other possible visibility settings are public and private.

Source File

- futu.sol

Locations

```
899  uint256 totalBuyTax = 3;  
900  uint256 totalSellTax = 3;  
901  
902  mapping (address => bool) public excludedFromFees;  
903
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 993

low SEVERITY

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

Source File

- futu.sol

Locations

```
992 address[] memory path = new address[](2);
993 path[0] = address(this);
994 path[1] = router.WETH();
995
996 _approve(address(this), address(router), tokenAmount);
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 994

low SEVERITY

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

Source File

- futu.sol

Locations

```
993   path[0] = address(this);  
994   path[1] = router.WETH();  
995  
996   _approve(address(this), address(router), tokenAmount);  
997
```

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.