InterFi NETWORK

@INTERFINETWORK

SMART CONTRACT SECURITY AUDIT OF

HEART TOKEN

SMART CONTRACT AUDIT | TEAM KYC | PROJECT EVALUATION

RELENTLESSLY SECURING THE PUBLIC BLOCKCHAIN | MADE IN CANADA

Summary

Auditing Firm	InterFi Network
Architecture	InterFi "Echelon" Auditing Standard
Smart Contract Audit Approved By	Chris | Blockchain Specialist at InterFi Network
Platform	Solidity
Audit Check (Mandatory)	Static, Software, Auto Intelligent & Manual Analysis
Project Check (Optional)	KYC
Consultation Request Date	November 03, 2021
Report Date	November 10, 2021

Audit Summary

InterFi team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

- **Heart Token's smart contract source code has LOW RISK SEVERITY.**
- **Heart Token has PASSED the smart contract audit.**

For the detailed understanding of risk severity, source code vulnerability, and functional test, kindly refer to the audit.

Table Of Contents

Project Information

Overview 4

InterFi "Echelon" Audit Standard

Audit Scope & Methodology 6

InterFi's Risk Classification 8

Smart Contract Risk Assessment

Static Analysis 9

Software Analysis 13

Manual Analysis 16

SWC Attacks 18

Risk Status & Radar Chart 20

Report Summary

Auditor's Verdict 21

Legal Advisory

Important Disclaimer 22

About InterFi Network 23

Project Overview

InterFi was consulted by Heart Token to conduct a smart contract security audit of their token source code.

About Heart Token (#cryptoforgood)

Heart Token (HEART) aims to build a community and an ecosystem that puts HEART into transforming the world for the better. Our ecosystem will bring accountability and protect investor's interest by implementing tools to improve transparency and governance. The Heart Token ecosystem will be an all-encompassing crypto platform that enables tokens to launch and find crypto related services safely and all in one place.

Project	Heart Token
Blockchain	Binance Smart Chain
Language	Solidity
Block Explorer	https://bscscan.com/address/0x5BEBE66053d22480E97fB4000Ed60c35fb203f49#code
Contract	0x5BEBE66053d22480E97fB4000Ed60c35fb203f49
Website	https://hearttoken.cc/
Telegram	https://t.me/hearttokenofficial
Twitter	http://twitter.com/hearttoken_
Instagram	https://instagram.com/_hearttoken_

Public logo

Solidity Source Code On Blockchain (BscScan Verified Contract Source Code)

https://bscscan.com/address/0x5BEBE66053d22480E97fB4000Ed60c35fb203f49#code

Contract Name: HeartToken (Token)

Compiler Version: v0.8.9+commit.e5eed63a

Optimization Enabled: Yes with 5000 runs

Solidity Source Code On InterFi GitHub

https://github.com/interfinetwork/audited-codes/blob/main/HeartToken.sol

InterFi GitHub Commit

Solidity source code is committed at: 3633199e8ee72774c4bc38300f112b86676hg7qk

SHA-1 Hash

Solidity source code is audited at hash #8a9cf02206e0d28cf4c305abf63725a5j4dc6dcc

Audit Scope & Methodology

The scope of this report is to audit the smart contract source code of Heart Token. The source code can be viewed in its entirety on

https://bscscan.com/address/0x5BEBE66053d22480E97fB4000Ed60c35fb203f49#code

InterFi has scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Category	
Smart Contract Vulnerabilities	❖ Re-entrancy (RE) ❖ Unhandled Exceptions (UE) ❖ Transaction Order Dependency (TO) ❖ Integer Overflow (IO) ❖ Unrestricted Action (UA) ❖ Ownership Takeover ❖ Gas Limit and Loops
Source Code Review	❖ Deployment Consistency ❖ Repository Consistency ❖ Data Consistency ❖ Token Supply Manipulation
Functional Assessment	❖ Access Control and Authorization ❖ Operations Trail and Event Generation ❖ Assets Manipulation ❖ Liquidity Access

InterFi's Echelon Audit Standard

The aim of InterFi's "Echelon" standard is to analyze the smart contract and identify the vulnerabilities and the hacks in the smart contract. Mentioned are the steps used by ECHELON-1 to assess the smart contract:

1. Solidity smart contract source code reviewal:
   - ❖ Review of the specifications, sources, and instructions provided to InterFi to make sure we understand the size, scope, and functionality of the smart contract.
   - ❖ Manual review of code, which is the process of reading source code line-byline to identify potential vulnerabilities.
2. Static, Manual, and Automated AI analysis:
   - ❖ Test coverage analysis, which is the process of determining whether the test cases are covering the code and how much code is exercised when we run those test cases.
   - ❖ Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts

Automated 3P frameworks used to assess the smart contract vulnerabilities

- ❖ Slither
- ❖ Consensys MythX
- ❖ Consensys Surya
- ❖ Open Zeppelin Code Analyzer
- ❖ Solidity Code Complier

InterFi's Risk Classification

Smart contracts are generally designed to manipulate and hold funds denominated in ETH/BNB. This makes them very tempting attack targets, as a successful attack may allow the attacker to directly steal funds from the contract. Below are the typical risk levels of a smart contract:

Vulnerable: A contract is vulnerable if it has been flagged by a static analysis tool as such. As we will see later, this means that some contracts may be vulnerable because of a false-positive.

Exploitable: A contract is exploitable if it is vulnerable and the vulnerability could be exploited by an external attacker. For example, if the "vulnerability" flagged by a tool is in a function which requires to own the contract, it would be vulnerable but not exploitable.

Exploited: A contract is exploited if it received a transaction on the main network which triggered one of its vulnerabilities. Therefore, a contract can be vulnerable or even exploitable without having been exploited.

Risk severity	Meaning
! Critical	This level vulnerabilities could be exploited easily, and can lead to asset loss, data loss, asset manipulation, or data manipulation. They should be fixed right away.
! High	This level vulnerabilities are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to critical risk severity
! Medium	This level vulnerabilities are should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution.
! Low	This level vulnerabilities can be ignored. They are code style violations, and informational statements in the code. They may not affect the smart contract execution

Smart Contract – Static Analysis

Symbol	Meaning
🛑	Function can be modified
💵	Function is payable
🔒	Function is locked
🔐	Function can be accessed
!	Important functionality

```
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |  | |
| └ | _msgData | Internal 🔒 |  | |
||||||
| **IERC20** | Interface |  |||
| └ | totalSupply | External ! |  |NO! |
| └ | decimals | External ! |  |NO! |
| └ | symbol | External ! |  |NO! |
| └ | name | External ! |  |NO! |
| └ | getOwner | External ! |  |NO! |
| └ | balanceOf | External ! |  |NO! |
| └ | transfer | External ! | 🛑 |NO! |
| └ | allowance | External ! |  |NO! |
| └ | approve | External ! | 🛑 |NO! |
| └ | transferFrom | External ! | 🛑 |NO! |
||||||
| **IUniswapV2Factory** | Interface |  |||
| └ | feeTo | External ! |  |NO! |
| └ | feeToSetter | External ! |  |NO! |
| └ | getPair | External ! |  |NO! |
| └ | allPairs | External ! |  |NO! |
| └ | allPairsLength | External ! |  |NO! |
| └ | createPair | External ! | 🛑 |NO! |
| └ | setFeeTo | External ! | 🛑 |NO! |
| └ | setFeeToSetter | External ! | 🛑 |NO! |
||||||
| **IUniswapV2Pair** | Interface |  |||
| └ | name | External ! |  |NO! |
| └ | symbol | External ! |  |NO! |
| └ | decimals | External ! |  |NO! |
| └ | totalSupply | External ! |  |NO! |
```

└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🛑	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗
└	swap	External ❗	🛑	NO❗
└	skim	External ❗	🛑	NO❗
└	sync	External ❗	🛑	NO❗
└	initialize	External ❗	🛑	NO❗
IUniswapV2Router01	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🛑	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🛑	NO❗
└	removeLiquidityETH	External ❗	🛑	NO❗
└	removeLiquidityWithPermit	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO❗
└	swapExactTokensForTokens	External ❗	🛑	NO❗
└	swapTokensForExactTokens	External ❗	🛑	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🛑	NO❗
└	swapExactTokensForETH	External ❗	🛑	NO❗
└	swapETHForExactTokens	External ❗	💵	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗

```
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO❗ |
||||||
| **AntiSnipe** | Interface |  |||
| └ | checkUser | External ❗ | 🛑 |NO❗ |
| └ | setLaunch | External ❗ | 🛑 |NO❗ |
| └ | setLpPair | External ❗ | 🛑 |NO❗ |
| └ | setProtections | External ❗ | 🛑 |NO❗ |
| └ | setGasPriceLimit | External ❗ | 🛑 |NO❗ |
| └ | removeSniper | External ❗ | 🛑 |NO❗ |
||||||
| **Cashier** | Interface |  |||
| └ | whomst | External ❗ |  |NO❗ |
| └ | setReflectionCriteria | External ❗ | 🛑 |NO❗ |
| └ | tally | External ❗ | 🛑 |NO❗ |
| └ | load | External ❗ | 💵 |NO❗ |
| └ | cashout | External ❗ | 🛑 |NO❗ |
| └ | giveMeWelfarePlease | External ❗ | 🛑 |NO❗ |
| └ | getTotalDistributed | External ❗ |  |NO❗ |
| └ | getShareholderInfo | External ❗ |  |NO❗ |
| └ | getShareholderRealized | External ❗ |  |NO❗ |
| └ | getPendingRewards | External ❗ |  |NO❗ |
| └ | initialize | External ❗ | 🛑 |NO❗ |
||||||
| **HeartToken** | Implementation | IERC20 |||
| └ | <Constructor> | Public ❗ | 💵 |NO❗ |
| └ | owner | Public ❗ |  |NO❗ |
| └ | transferOwner | External ❗ | 🛑 | onlyOwner |
| └ | renounceOwnership | Public ❗ | 🛑 | onlyOwner |
| └ | <Receive Ether> | External ❗ | 💵 |NO❗ |
| └ | totalSupply | External ❗ |  |NO❗ |
| └ | decimals | External ❗ |  |NO❗ |
| └ | symbol | External ❗ |  |NO❗ |
| └ | name | External ❗ |  |NO❗ |
| └ | getOwner | External ❗ |  |NO❗ |
| └ | balanceOf | Public ❗ |  |NO❗ |
| └ | allowance | External ❗ |  |NO❗ |
| └ | approve | Public ❗ | 🛑 |NO❗ |
| └ | approveMax | Public ❗ | 🛑 |NO❗ |
| └ | _approve | Private 🔐 | 🛑 | |
| └ | transfer | External ❗ | 🛑 |NO❗ |
| └ | transferFrom | External ❗ | 🛑 |NO❗ |
| └ | isFeeExcluded | Public ❗ |  |NO❗ |
| └ | isDividendExcluded | Public ❗ |  |NO❗ |
| └ | isMaxWalletExcluded | Public ❗ |  |NO❗ |
| └ | setInitializers | External ❗ | 🛑 | onlyOwner |
| └ | removeSniper | External ❗ | 🛑 | onlyOwner |
| └ | setProtectionSettings | External ❗ | 🛑 | onlyOwner |
| └ | setGasPriceLimit | External ❗ | 🛑 | onlyOwner |
| └ | enableTrading | Public ❗ | 🛑 | onlyOwner |
| └ | setDividendExcluded | Public ❗ | 🛑 | onlyOwner |
```

```
| └ | setMaxWalletExcluded | Public ❗ | 🛑 | onlyOwner |
| └ | setExcludedFromFees | Public ❗ | 🛑 | onlyOwner |
| └ | setTaxes | External ❗ | 🛑 | onlyOwner |
| └ | setRatios | External ❗ | 🛑 | onlyOwner |
| └ | setWallets | External ❗ | 🛑 | onlyOwner |
| └ | setContractSwapSettings | External ❗ | 🛑 | onlyOwner |
| └ | setSwapSettings | External ❗ | 🛑 | onlyOwner |
| └ | setReflectionCriteria | External ❗ | 🛑 | onlyOwner |
| └ | setReflectorSettings | External ❗ | 🛑 | onlyOwner |
| └ | giveMeWelfarePlease | External ❗ | 🛑 | NO❗ |
| └ | getTotalReflected | External ❗ | | NO❗ |
| └ | getUserInfo | External ❗ | | NO❗ |
| └ | getUserRealizedGains | External ❗ | | NO❗ |
| └ | getUserUnpaidEarnings | External ❗ | | NO❗ |
| └ | setNewRouter | Public ❗ | 🛑 | onlyOwner |
| └ | setLpPair | External ❗ | 🛑 | onlyOwner |
| └ | setMaxWalletSize | Public ❗ | 🛑 | onlyOwner |
| └ | excludePresaleAddresses | External ❗ | 🛑 | onlyOwner |
| └ | _hasLimits | Private 🔐 | | |
| └ | _transfer | Internal 🔒 | 🛑 | |
| └ | _finalizeTransfer | Internal 🔒 | 🛑 | |
| └ | processTokenReflect | Internal 🔒 | 🛑 | |
| └ | _basicTransfer | Internal 🔒 | 🛑 | |
| └ | takeTaxes | Internal 🔒 | 🛑 | |
| └ | contractSwap | Internal 🔒 | 🛑 | swapping |
| └ | manualDepost | External ❗ | 🛑 | onlyOwner |
| └ | _checkLiquidityAdd | Private 🔐 | 🛑 | |
```

Smart Contract – Software Analysis

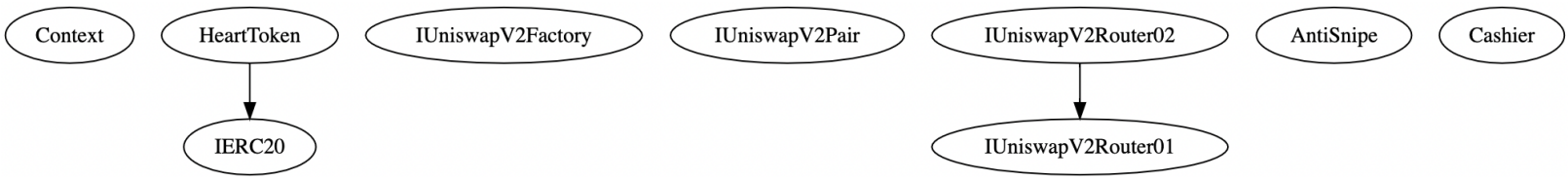
Function Signatures

```
26003957  =>  setMaxWalletSize(uint256,uint256)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
18160ddd  =>  totalSupply()
313ce567  =>  decimals()
95d89b41  =>  symbol()
06fdde03  =>  name()
893d20e8  =>  getOwner()
70a08231  =>  balanceOf(address)
a9059cbb  =>  transfer(address,uint256)
dd62ed3e  =>  allowance(address,address)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
017e7e58  =>  feeTo()
094b7415  =>  feeToSetter()
e6a43905  =>  getPair(address,address)
1e3dd18b  =>  allPairs(uint256)
574f2ba3  =>  allPairsLength()
c9c65396  =>  createPair(address,address)
f46901ed  =>  setFeeTo(address)
a2e74af6  =>  setFeeToSetter(address)
3644e515  =>  DOMAIN_SEPARATOR()
30adf81f  =>  PERMIT_TYPEHASH()
7ecebe00  =>  nonces(address)
d505accf  =>  permit(address,address,uint256,uint256,uint8,bytes32,bytes32)
ba9a7a56  =>  MINIMUM_LIQUIDITY()
c45a0155  =>  factory()
0dfe1681  =>  token0()
d21220a7  =>  token1()
0902f1ac  =>  getReserves()
5909c0d5  =>  price0CumulativeLast()
5a3d5493  =>  price1CumulativeLast()
7464fc3d  =>  kLast()
6a627842  =>  mint(address)
89afcb44  =>  burn(address)
022c0d9f  =>  swap(uint256,uint256,address,bytes)
bc25cf77  =>  skim(address)
fff6cae9  =>  sync()
485cc955  =>  initialize(address,address)
ad5c4648  =>  WETH()
e8e33700  =>  addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256)
f305d719  =>  addLiquidityETH(address,uint256,uint256,uint256,address,uint256)
baa2abde  =>  removeLiquidity(address,address,uint256,uint256,uint256,address,uint256)
02751cec  =>  removeLiquidityETH(address,uint256,uint256,uint256,address,uint256)
```

```
2195995c  =>
removeLiquidityWithPermit(address,address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes3
2,bytes32)
ded9382a  =>
removeLiquidityETHWithPermit(address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes32,byt
es32)
38ed1739  =>  swapExactTokensForTokens(uint256,uint256,address[],address,uint256)
8803dbee  =>  swapTokensForExactTokens(uint256,uint256,address[],address,uint256)
7ff36ab5  =>  swapExactETHForTokens(uint256,address[],address,uint256)
4a25d94a  =>  swapTokensForExactETH(uint256,uint256,address[],address,uint256)
18cbafe5  =>  swapExactTokensForETH(uint256,uint256,address[],address,uint256)
fb3bdb41  =>  swapETHForExactTokens(uint256,address[],address,uint256)
ad615dec  =>  quote(uint256,uint256,uint256)
054d50d4  =>  getAmountOut(uint256,uint256,uint256)
85f8c259  =>  getAmountIn(uint256,uint256,uint256)
d06ca61f  =>  getAmountsOut(uint256,address[])
1f00ca74  =>  getAmountsIn(uint256,address[])
af2979eb  =>
removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,uint256,uint256,address,uint256)
5b0d5984  =>
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens(address,uint256,uint256,uint256,address,u
int256,bool,uint8,bytes32,bytes32)
5c11d795  =>
swapExactTokensForTokensSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)
b6f9de95  =>  swapExactETHForTokensSupportingFeeOnTransferTokens(uint256,address[],address,uint256)
791ac947  =>
swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)
ecd07a53  =>  checkUser(address,address,uint256)
4b38f1d4  =>  setLaunch(address,uint32,uint64)
80c581d1  =>  setLpPair(address,bool)
0109da69  =>  setProtections(bool,bool,bool,bool)
09231602  =>  setGasPriceLimit(uint256)
33251a0b  =>  removeSniper(address)
c8a4a3de  =>  whomst()
8cd7f02e  =>  setReflectionCriteria(uint256,uint256)
df047e9e  =>  tally(address,uint256)
86d5c4be  =>  load()
a9e732bb  =>  cashout(uint256)
2e6f0efd  =>  giveMeWelfarePlease(address)
5695fa58  =>  getTotalDistributed()
5e6056bb  =>  getShareholderInfo(address)
d59b2ffd  =>  getShareholderRealized(address)
f6ed2017  =>  getPendingRewards(address)
8129fc1c  =>  initialize()
8da5cb5b  =>  owner()
4fb2e45d  =>  transferOwner(address)
715018a6  =>  renounceOwnership()
571ac8b0  =>  approveMax(address)
104e81ff  =>  _approve(address,address,uint256)
bca238aa  =>  isFeeExcluded(address)
```


```
255a2e54  =>  isDividendExcluded(address)
203a421f  =>  isMaxWalletExcluded(address)
218950bc  =>  setInitializers(address,address)
29dd8798  =>  setProtectionSettings(bool,bool,bool,bool)
8a8c523c  =>  enableTrading()
6b639a25  =>  setDividendExcluded(address,bool)
ae0f1cc0  =>  setMaxWalletExcluded(address,bool)
590ffdce  =>  setExcludedFromFees(address,bool)
32cde664  =>  setTaxes(uint16,uint16,uint16)
0712d165  =>  setRatios(uint16,uint16,uint16,uint16)
d3f6a157  =>  setWallets(address,address)
dea528a1  =>  setContractSwapSettings(bool,bool)
fb78680d  =>  setSwapSettings(uint256,uint256,uint256,uint256)
09a8f179  =>  setReflectionCriteria(uint256,uint256,uint256)
d0b1449d  =>  setReflectorSettings(uint256)
9dc67615  =>  giveMeWelfarePlease()
f7c245ea  =>  getTotalReflected()
6386c1c7  =>  getUserInfo(address)
6372c68f  =>  getUserRealizedGains(address)
ab5cb5a6  =>  getUserUnpaidEarnings(address)
eafb5a3c  =>  setNewRouter(address)
13b4a7f4  =>  excludePresaleAddresses(address,address)
6d6b6f11  =>  _hasLimits(address,address)
30e0789e  =>  _transfer(address,address,uint256)
0c34ebd5  =>  _finalizeTransfer(address,address,uint256,bool)
20915ab5  =>  processTokenReflect(address,address)
f0774e71  =>  _basicTransfer(address,address,uint256)
aa0ffca5  =>  takeTaxes(address,address,uint256)
764cdf3d  =>  contractSwap(uint256)
f7727e14  =>  manualDepost()
8c234eb0  =>  _checkLiquidityAdd(address,address)
```

Inheritance Graph



Smart Contract – Manual Analysis

Function	Description	Tested	Verdict
Total Supply	provides information about the total token supply	Yes	**Passed**
Balance Of	provides account balance of the owner's account	Yes	**Passed**
Transfer	executes transfers of a specified number of tokens to a specified address	Yes	**Passed**
Approve	allow a spender to withdraw a set number of tokens from a specified account	Yes	**Passed**
Allowance	returns a set number of tokens from a spender to the owner	Yes	**Passed**
Buy Back	is an action in which the project buys back its tokens from the existing holders usually at a market price	NA	NA
Burn	executes transfers of a specified number of tokens to a burn address	NA	NA
Mint	executes creation of a specified number of tokens and adds it to the total supply	NA	NA
Rebase	circulating token supply adjusts (increases or decreases) automatically according to a token's price fluctuations	NA	NA
Blacklist	stops specified wallets from interacting with the smart contract function modules	NA	NA
Lock	stops or locks all function modules of the smart contract	NA	NA

Important Information

- Active Owner: 0x5789ae9ebe7bc65f25c3ba5e9d1e52f2aa9415ba
- **Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security.**
- Owner can-not lock or burn user assets.
- Owner can-not stop or pause the smart contract.
- **Heart Token utilizes Anti-Snipe function modules. This module may stop certain wallets from trading abruptly. (Used to stop trading bots)**

```
    interface AntiSnipe {
    function checkUser(address from, address to, uint256 amt) external returns (bool);
    function setLaunch(address _initialLpPair, uint32 _liqAddBlock, uint64 _liqAddStamp) external;
    function setLpPair(address pair, bool enabled) external;
    function setProtections(bool _as, bool _ag, bool _ab, bool _aspecial) external;
    function setGasPriceLimit(uint256 gas) external;
    function removeSniper(address account) external;
}
```

- **Heart Token smart contract had mentioned fees and wallet at the time of the audit, please note these fees and wallets can be changed by the active owner.**


```
  Ratios public _ratios = Ratios({
       rewards: 5,
       liquidity: 2,
       marketing: 3,
       charity: 3,
       total: 13
       });

    // PCS ROUTER
    address private _routerAddress = 0x10ED43C718714eb63d5aA57B78B54704E256024E;

    address private WBNB;

    address payable private _marketingWallet = payable(0xf36E4c86a57B4694F4E621F0936f7bEFE8bbAcAE);
    address payable private _charityWallet = payable(0x77fFfcdA205488c1424B57Ae34b9181A43252D3E);
```

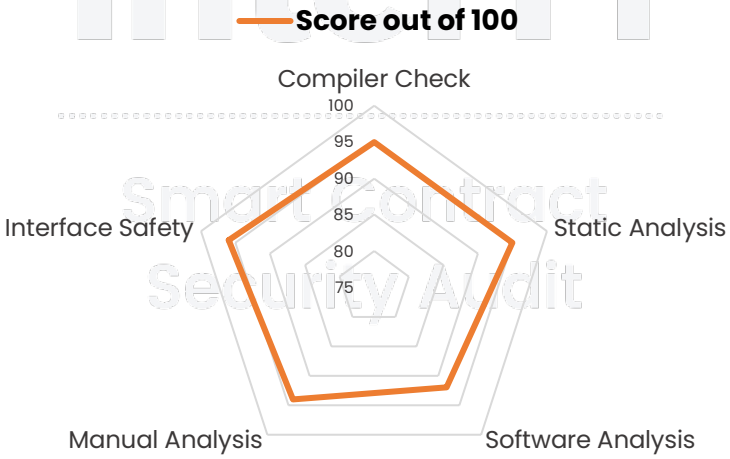
Smart Contract – SWC Attacks

SWC ID	Description	Verdict
SWC-101	Integer Overflow and Underflow	**Passed**
SWC-102	Outdated Compiler Version	**Passed**
SWC-103	Floating Pragma	**Passed**
SWC-104	Unchecked Call Return Value	**Passed**
SWC-105	Unprotected Ether Withdrawal	**Passed**
SWC-106	Unprotected SELFDESTRUCT Instruction	**Passed**
SWC-107	Re-entrancy	**Passed**
SWC-108	State Variable Default Visibility	**Passed**
SWC-109	Uninitialized Storage Pointer	**Passed**
SWC-110	Assert Violation	**Passed**
SWC-111	Use of Deprecated Solidity Functions	**Passed**
SWC-112	Delegate Call to Untrusted Callee	**Passed**
SWC-113	DoS with Failed Call	**Passed**
SWC-114	Transaction Order Dependence	**Passed**
SWC-115	Authorization through tx.origin	**Passed**
SWC-116	Block values as a proxy for time	**Passed**
SWC-117	Signature Malleability	**Passed**
SWC-118	Incorrect Constructor Name	**Passed**

SWC-119	Shadowing State Variables	**Passed**
SWC-120	Weak Sources of Randomness from Chain Attributes	**Passed**
SWC-121	Missing Protection against Signature Replay Attacks	**Passed**
SWC-122	Lack of Proper Signature Verification	**Passed**
SWC-123	Requirement Violation	**Passed**
SWC-124	Write to Arbitrary Storage Location	**Passed**
SWC-125	Incorrect Inheritance Order	**Passed**
SWC-126	Insufficient Gas Griefing	**Passed**
SWC-127	Arbitrary Jump with Function Type Variable	**Passed**
SWC-128	DoS With Block Gas Limit	**Passed**
SWC-129	Typographical Error	**! Low**
SWC-130	Right-To-Left-Override control character (U+202E)	**Passed**
SWC-131	Presence of unused variables	**Passed**
SWC-132	Unexpected Ether balance	**Passed**
SWC-133	Hash Collisions With Multiple Variable Length Arguments	**Passed**
SWC-134	Message call with hardcoded gas amount	**Passed**
SWC-135	Code With No Effects (Irrelevant/Dead Code)	**! Low**
SWC-136	Unencrypted Private Data On-Chain	**Passed**

Smart Contract - Risk Status & Radar Chart

Risk Severity	Status
! Critical	None critical severity issues identified
! High	None high severity issues identified
! Medium	None medium severity issues identified
! Low	None low severity issues identified
Passed	**41 functions and instances verified and passed**



Compiler Check	95
Static Analysis	95
Software Analysis	92
Manual Analysis	94
Interface Safety	96

Auditor's Verdict

InterFi team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks.

Heart Token's smart contract source code has LOW RISK SEVERITY.

Heart Token has PASSED the smart contract audit.

Auditor's Note:

- Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security.
- Project's liquidity pair isn't checked and verified due to out of scope.
- Project website is not checked due to out of scope. The website hasn't been reviewed for SSL and lighthouse report.

Important Disclaimer

InterFi Network provides contract auditing and project verification services for blockchain projects. The purpose of the audit is to analyse the on-chain smart contract source code, and to provide basic overview of the project. **This report should not be transmitted, disclosed, referred to, or relied upon by any person for any purposes without InterFi's prior written consent.**

InterFi provides the easy-to-understand assessment of the project, and the smart contract (otherwise known as the source code). The audit makes no statements or warranties on the security of the code. It also cannot be considered as an enough assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. While we have used all the data at our disposal to provide the transparent analysis, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. **Be aware that smart contracts deployed on a blockchain aren't resistant from external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security. Therefore, InterFi does not guarantee the explicit security of the audited smart contract.**

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

This report should not be considered as an endorsement or disapproval of any project or team. The information provided on this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. Do conduct your own due diligence and consult your financial advisor before making any investment decisions.

About InterFi Network

InterFi Network provides intelligent blockchain solutions. InterFi is developing an ecosystem that is seamless and responsive. Some of our services: Blockchain Security, Token Launchpad, NFT Marketplace, etc. **InterFi's mission is to interconnect multiple services like Blockchain Security, DeFi, Gaming, and Marketplace under one ecosystem that is seamless, multi-chain compatible, scalable, secure, fast, responsive, and easy-to-use.**

InterFi is built by a decentralized team of UI experts, contributors, engineers, and enthusiasts from all over the world. Our team currently consists of 6+ core team members, and 10+ casual contributors. **InterFi provides manual, static, and automatic smart contract analysis, to ensure that project is checked against known attacks and potential vulnerabilities.**

To learn more, visit https://interfi.network

To view our audit portfolio, visit https://github.com/interfinetwork

To book an audit, message https://t.me/interfiaudits

@INTERFINETWORK

RELENTLESSLY SECURING THE PUBLIC BLOCKCHAIN | MADE IN CANADA