



SMART CONTRACT AUDIT

 interfinetwork

 hello@interfi.network

 <https://interfi.network>

PREPARED FOR

SHINA INU STAKE CONTRACT



INTRODUCTION

Auditing Firm	InterFi Network
Client Firm	Shina Inu
Methodology	Automated Analysis, Manual Code Review
Language	Solidity
Contract#1	0xF0bfC0246109dF3CFA02880fFe7f0b8933a86Dc7
Contract#2	0xcd5d2cde7feddbfcabdbedbcc0342b060493fc1c
Blockchain	Ethereum
Centralization	Active Ownership
Commit	2d3560f34da223ca413f0ab63e8ebc999ed96881
Website	https://shinatoken.com/
Telegram	https://t.me/newShinaTokenPortal
X (Twitter)	https://twitter.com/ShinaToken
Report Date	November 23, 2024


 Verify the authenticity of this report on our website: <https://www.github.com/interfinetwork>



EXECUTIVE SUMMARY

InterFi has performed the automated and manual analysis of solidity codes. Solidity codes were reviewed for common contract vulnerabilities and centralized exploits. Here's a quick audit summary:

Status	Critical ●	Major ●	Medium ●	Minor ●	Unknown ●
Open	0	0	0	4	0
Acknowledged	0	2	1	1	2
Resolved	0	1	1	1	0
Important Functions	rwd, give, withdrawOCFee, stake, withdraw, reset vote, vote, create				
Noteworthy Privileges	setEpochInterval, setPool, withdrawTkn, setConsensusReq, setOCFee, setDest, setMaxBurnPrc, setBurnFactor, setDonationPrc				

 Please note that smart contracts deployed on blockchains aren't resistant to exploits, vulnerabilities and/or hacks. Blockchain and cryptography assets utilize new and emerging technologies. These technologies present a high level of ongoing risks. For a detailed understanding of risk severity, source code vulnerability, and audit limitations, kindly review the audit report thoroughly.


 Please note that centralization privileges regardless of their inherited risk status – constitute an elevated impact on smart contract safety and security.



TABLE OF CONTENTS


TABLE OF CONTENTS	4
SCOPE OF WORK.....	5
AUDIT METHODOLOGY	6
RISK CATEGORIES	8
CENTRALIZED PRIVILEGES	9
AUTOMATED ANALYSIS.....	10
INHERITANCE GRAPH	13
MANUAL REVIEW	14
DISCLAIMERS	31
ABOUT INTERFI NETWORK	34



SCOPE OF WORK

InterFi was consulted to conduct the smart contract audit of mentioned solidity source codes. The audit scope of work is strictly limited to these solidity file(s) only:

- Ktv2.sol
- Ktv2Factory.sol

 If source codes are not deployed on the main net, they can be modified or altered before main-net deployment. Verify the contract's deployment status below:

Public Contract Link	
https://etherscan.io/address/0xF0bfC0246109dF3CFA02880fFe7f0b8933a86Dc7#code https://etherscan.io/address/0xcd5d2cde7feddbfcabdbedbcc0342b060493fc1c#code	
Compiler Version	0.8.16
License	Unlicensed



AUDIT METHODOLOGY

Smart contract audits are conducted using a set of standards and procedures. Mutual collaboration is essential to performing an effective smart contract audit. Here's a brief overview of InterFi's auditing process and methodology:

CONNECT

- The onboarding team gathers source codes, and specifications to make sure we understand the size, and scope of the smart contract audit.

AUDIT

- Automated analysis is performed to identify common contract vulnerabilities. We may use the following third-party frameworks and dependencies to perform the automated analysis:
 - Remix IDE Developer Tool
 - Open Zeppelin Code Analyzer
 - SWC Vulnerabilities Registry
 - DEX Dependencies, e.g., Pancakeswap, Uniswap
- Simulations are performed to identify centralized exploits causing contract and/or trade locks.
- A manual line-by-line analysis is performed to identify contract issues and centralized privileges.

We may inspect below mentioned common contract vulnerabilities, and centralized exploits:

Centralized Exploits	<ul style="list-style-type: none"> ○ Token Supply Manipulation ○ Access Control and Authorization ○ Assets Manipulation ○ Ownership Control ○ Liquidity Access ○ Stop and Pause Trading ○ Ownable Library Verification
----------------------	---



Common Contract Vulnerabilities	<ul style="list-style-type: none"> ○ Integer Overflow ○ Lack of Arbitrary limits ○ Incorrect Inheritance Order ○ Typographical Errors ○ Requirement Violation ○ Gas Optimization ○ Coding Style Violations ○ Re-entrancy ○ Third-Party Dependencies ○ Potential Sandwich Attacks ○ Irrelevant Codes ○ Divide before multiply ○ Conformance to Solidity Naming Guides ○ Compiler Specific Warnings ○ Language Specific Warnings
---------------------------------	---

REPORT

- The auditing team provides a preliminary report specifying all the checks which have been performed and the findings thereof.
- The client's development team reviews the report and makes amendments to solidity codes.
- The auditing team provides the final comprehensive report with open and unresolved issues.

PUBLISH

- The client may use the audit report internally or disclose it publicly.

 It is important to note that there is no pass or fail in the audit, it is recommended to view the audit as an unbiased assessment of the safety of solidity codes.



RISK CATEGORIES

A successful external attack may allow the external attacker to directly exploit. A successful centralization-related exploit may allow the privileged role to directly exploit. All risks which are identified in the audit report are categorized:

Risk Type	Definition
Critical ●	These risks pose immediate and severe threats, such as asset theft, data manipulation, or complete loss of contract functionality. They are often easy to exploit and can lead to significant, irreparable damage. Immediate fix is required.
Major ●	These risks can significantly impact code performance and security, and they may indirectly lead to asset theft and data loss. They can allow unauthorized access or manipulation of sensitive functions if exploited. Fixing these risks are important.
Medium ●	These risks may create attack vectors under certain conditions. They may enable minor unauthorized actions or lead to inefficiencies that can be exploited indirectly to escalate privileges or impact functionality over time.
Minor ●	These risks may include inefficiencies, lack of optimizations, code-style violations. These should be addressed to enhance overall code quality and maintainability.
Unknown ●	These risks pose uncertain severity to the contract or those who interact with it. Immediate fix is required to mitigate risk uncertainty.

All statuses which are identified in the audit report are categorized here:

Status Type	Definition
Open	Risks are open.
Acknowledged	Risks are acknowledged, but not fixed.
Resolved	Risks are acknowledged and fixed.



CENTRALIZED PRIVILEGES

Centralization risk is the most common cause of cryptography asset loss. When a smart contract has a privileged role, the risk related to centralization is elevated.

There are some well-intended reasons have privileged roles, such as:

- Privileged roles can be granted the power to pause() the contract in case of an external attack.
- Privileged roles can use functions like, include(), and exclude() to add or remove wallets from fees, swap checks, and transaction limits. This is useful to run a presale and to list on an exchange.

Authorizing privileged roles to externally-owned-account (EOA) is dangerous. Lately, centralization-related losses are increasing in frequency and magnitude.

- The client can lower centralization-related risks by implementing below mentioned practices:
- Privileged role's private key must be carefully secured to avoid any potential hack.
- Privileged role should be shared by multi-signature (multi-sig) wallets.
- Authorized privilege can be locked in a contract, user voting, or community DAO can be introduced to unlock the privilege.
- Renouncing the contract ownership, and privileged roles.
- Remove functions with elevated centralization risk.

 Understand the project's initial asset distribution. Assets in the liquidity pair should be locked. Assets outside the liquidity pair should be locked with a release schedule.



AUTOMATED ANALYSIS

Symbol	Definition
	Function modifies state
	Function is payable
	Function is internal
	Function is private
!	Function is important

```

| **Ktv2Factory** | Implementation | |||
| L | create | External ! |  | NO ! |
| L | count | Public ! | | NO ! |
|||||
| **TPI** | Interface | |||
| L | price | External ! | | NO ! |
|||||
| **Ktv2** | Implementation | Ownable |||
| L | <Constructor> | Public ! |  | Ownable |
| L | <Receive Ether> | External ! |  | NO ! |
| L | rwd | External ! |  | onlyOC notDeclined epochComplete |
| L | vote | External ! |  | onlyOC notDeclined epochComplete |
| L | resetVote | External ! |  | onlyOC notDeclined epochComplete |
| L | resetOCFee | Private  |  | |
| L | recordOCFee | Private  |  | |
| L | withdrawOCFee | External ! |  | NO ! |
| L | setEpochInterval | Public ! |  | onlyOwner |
| L | setConsensusReq | Public ! |  | onlyOwner |

```

INTERFI
CONFIDENTIAL



```

|  L | setOCFee | Public ! | ● | onlyOwner |
|  L | addOCRwdr | Public ! | ● | onlyOwner |
|  L | removeOCRwdr | Public ! | ● | onlyOwner |
|  L | setDest | Public ! | ● | onlyOwner |
|  L | setMaxBurnPrc | Public ! | ● | onlyOwner |
|  L | setBurnFactor | Public ! | ● | onlyOwner |
|  L | setDonationPrc | Public ! | ● | onlyOwner |
|  L | setPool | Public ! | ● | onlyOwner |
|  L | decline | Public ! | ● | NO ! |
|  L | allow | Public ! | ● | NO ! |
|  L | stake | External ! | ● | NO ! |
|  L | withdraw | External ! | ● | NO ! |
|  L | give | External ! | 🚫 | NO ! |
|  L | withdrawTkn | External ! | ● | onlyOwner |
|||||
| **Ownable** | Implementation | Context |||
|  L | <Constructor> | Public ! | ● | NO ! |
|  L | owner | Public ! | | NO ! |
|  L | _checkOwner | Internal 🔒 | | |
|  L | renounceOwnership | Public ! | ● | onlyOwner |
|  L | transferOwnership | Public ! | ● | onlyOwner |
|  L | _transferOwnership | Internal 🔒 | ● | |
|||||
| **IERC20** | Interface | |||
|  L | totalSupply | External ! | | NO ! |
|  L | balanceOf | External ! | | NO ! |
|  L | transfer | External ! | ● | NO ! |
|  L | allowance | External ! | | NO ! |

```

INTERFI
CONFIDENTIAL

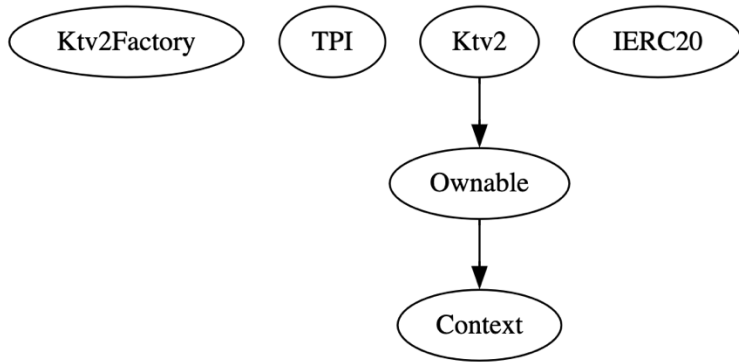
```
| ^ | approve | External ! | 🚫 | NO ! |  
| ^ | transferFrom | External ! | 🚫 | NO ! |  
|||||  
| **Context** | Implementation | |||  
| ^ | _msgSender | Internal 🔒 | | |  
| ^ | _msgData | Internal 🔒 | | |  
| ^ | _contextSuffixLength | Internal 🔒 | | |
```

INTERFI
CONFIDENTIAL

INTERFI
CONFIDENTIAL



INHERITANCE GRAPH



INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL



MANUAL REVIEW

Identifier	Definition	Severity
CEN-01	Centralized privileges	Major 🟡
CEN-01-01	Privileged role has authority to set fees	
CEN-01-02	Privileged role can set pool	
CEN-07	Authorizations and access control	

Important onLy0wner centralized privileges are listed below:

setEpochInterval
 setConsensusReq
 setOCFee
 addOCRwdr
 removeOCRwdr
 setDest
 setMaxBurnPrc
 setBurnFactor
 setDonationPrc
 setPool
 withdrawTkn
 renounceOwnership
 transferOwnership

Noteworthy onLy0C controlled privileges are listed below:

rwd
 vote
 resetVote



RECOMMENDATION

Securing private keys or access credentials of deployers, contract owners, OC Rewarders (ocRwdrs), and other roles with privileged access is crucial to prevent single points of failure that can compromise contract security.

Use of multi-signature wallets is recommended – These wallets require multiple authorizations to execute sensitive contract functions, reducing the risk associated with single-party control.

Use of decentralized governance model is recommended – This model allows token holders and stakeholders to actively participate in decision-making, such as contract upgrades and parameter adjustments, enhancing overall security and resilience.

ACKNOWLEDGEMENT

Shina Inu team argued that centralized and controlled privileges are used as required.



Identifier	Definition	
CEN-03	Lack of circuit breaker	

Ktv2 and Ktv2Factory contracts currently lack a circuit breaker or emergency stop mechanism. Circuit breakers are crucial for halting contract functionality in response to active security breaches or critical bugs, allowing administrators to freeze all non-essential activities until issues are resolved.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

NOTE

Use a circuit breaker mechanism with a simple state variable that can disable critical functionalities such as token transfers, staking, withdrawals, and reward distribution when activated.



Identifier	Definition	Severity
CEN-04	Manipulation of consensus through single OC	Medium ●

The consensus mechanism in rwd function can be manipulated if a single OC controls multiple addresses or if there is collusion among a small number of OCs, as the consensus requirement can be set very low.


RECOMMENDATION

Use robust consensus mechanisms that require participation from a wider range of addresses and consider additional validation of OC actions to mitigate the risk of collusion.

ACKNOWLEDGEMENT

Shina Inu team argued that consensus mechanism is set as per required logic. Code pertaining to consensus mechanisms is kept as-is.



Identifier	Definition	Severity
LOG-01	Insufficient input boundaries	Minor 
LOG-01-02	Critical parameter changes by owner	

Below mentioned functions are set without sufficient input boundaries:

setEpochInterval

setOCFee

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

RECOMMENDATION

Establish clear upper boundaries. All operational parameters remain within safe and rational ranges. Changes should possibly go through a timelock and/or be subject to community voting to prevent misuse.



Identifier	Definition	Severity
LOG-02	Potential front-running	Minor 

Front-running is a concern in Ktv2 contract, especially in functions that involve financial transactions such as stake, withdraw, and give. Since these transactions depend on external factors like token prices and contract states, totalStk and totalBurned, they can be susceptible to front-running where an attacker sees a pending transaction and issues another transaction with a higher gas fee to get it mined first.

RECOMMENDATION

Functions that execute critical state changes should enforce minimum output thresholds. Setting these minimums above zero can deter malicious actors by reducing the predictability and profitability of front-running strategies.

Implement commit-reveal schemes or transaction ordering to protect against front-running.

ACKNOWLEDGEMENT

Front-running is not avoidable on public blockchains. Shina Inu team commented that, most EVM chains are prone to some sort of front-running and external manipulation.



Identifier	Definition	Severity
LOG-03	Re-entrancy	Major 🟡
LOG-04	Checks-Effects-Interactions	

Below mentioned functions are used without Re-entrancy guard:

```
rwd
give
withdraw0CFee
```

These functions in Ktv2 contract use calls to external addresses `to.call{value: _rwd}("")` and `msg.sender.call{value: amt}("")` which can be exploited. If the external address is a contract, it can execute fallback functions that call back into Ktv2 contract, leading to re-entrancy attacks where state changes can be manipulated before they are finalized.

Since the \$SHI token contract is verified on blockchain and treated as a trusted black-box for this audit, all interactions with \$SHI token contract within stake and withdraw functions are deemed valid and safe.

RECOMMENDATION

Use Checks-Effects-Interactions (CEI) pattern when transferring control to external entities. This design pattern ensures that all state changes are completed before external interactions occur. Additionally, implement re-entrancy guard to block recursive calls from external contracts.



CLIENT COMMENTS (ACKNOWLEDGED)


Regarding `rwd` and `withdrawOCFee` Functions:

Shina Inu team has acknowledged that these functions interact with an external contract, specifically OC contract, which introduces potential risks if the external contract is malicious. However, Shina Inu team has clarified that the OC contract will be off-chain and is currently under development. Once ready, that smart contract will undergo a comprehensive audit to ensure it is secure and free from vulnerabilities, including re-entrancy issues.

Regarding `give` Function:

Shina Inu team has clarified that `give` function does not interact with any external smart contract. Instead, it directly transfers ETH to `_dest` address, which is hardcoded as `0x750EF1D7a0b4Ab1c97B7A623D7917CcEb5ea779C`. This address is the official donation wallet of the *GiveDirectly* charity, which can be verified via the staking contract code on Etherscan and the [GiveDirectly](#) website.



Identifier	Definition	Severity
LOG-05	Lack of rate limiting	Minor 
LOG-06	Potential denial of service (DoS)	

Mentioned functions do not have any rate limiting mechanisms. This can potentially allow an OC to flood the system with votes, resets, or rewards within a short period, leading to denial of service (DoS) by clogging the system:

vote

resetVote

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT

RECOMMENDATION

Use rate limiting or cooldown periods for sensitive actions to prevent abuse.



Identifier	Definition	Severity
COD-03	Insufficient validation in token transfers	Medium 🟡

stake and withdraw functions transfer tokens based on ERC20 interface but do not handle possible failures of these transfers robustly. `require(token.transferFrom(msg.sender, address(this), amt), "Transfer failed");` and similar lines assume that transfer will always succeed if conditions are met.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

RECOMMENDATION

Enhance error handling around token transfers. Check return values more comprehensively.

RESOLUTION

Since \$SHI token contract adheres to ERC20 standard, require statements in stake and withdraw functions will correctly revert the transaction if transfer fails for any reason (e.g., insufficient allowance, insufficient balance, or invalid addresses).



Identifier	Definition	Severity
COD-04	Unchecked external calls	Major 🟡

Smart contract uses external calls `dest.call{value: giveAmt}("")` which can fail silently without reverting the transaction. This can be exploited by attackers or result in unintended loss of funds.

give

RECOMMENDATION


Use safer alternatives like `.send()` or `.transfer()`, which revert automatically on failure.

RESOLUTION

give function uses an external call `dest.call{value: giveAmt}("")` to transfer Ether. `require` statement is used to log failure.

While the `require(sent, "Donate failed")` ensures that transaction reverts on failure, the use of `.call()` can still introduce re-entrancy risks as mentioned in LOG-03. To mitigate these risks, implement re-entrancy guard.



Identifier	Definition	Severity
COD-05	Missing zero address validation	Minor 

Below mentioned functions are missing zero address input validation:

addOCRwdr

removeOCRwdr

setDest

setPool

withdrawTkn

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

RECOMMENDATION

Validate if the modified address is dead(0) or not.



Identifier	Definition	Severity
COD-06	Potential flash loan attack	Unknown 🟤

Smart contract function `rwd` that rely on external price feeds can be manipulated through flash loan attacks. There's an external price feed `tp.price(pool)`. When functions relied on these feeds for critical decision-making, and these feeds can be influenced by market manipulation, via flash loans.

Unless the attacker can influence the status of `ocRwdr` or manipulate the consensus directly, the susceptibility to a flash loan attack in this function is low.

RECOMMENDATION

To mitigate flash loan attacks, use checks that assess the duration for which a voter has held tokens before allowing voting rights. Also, use multiple price feeds to ascertain real-time asset values.

ACKNOWLEDGEMENT

Shina Inu team has confirmed that `ocRwdr` accounts will be authorized by the established consensus logic. As a result, they deem this method to be secure and have decided to maintain the current implementation without changes.



Identifier	Definition	Severity
COD-10	Direct and indirect dependencies	Unknown 🟡

Ktv2 and Ktv2Factory smart contracts interact with several third-party protocols and external contracts, including decentralized exchange (DEX) interfaces, token price interfaces (TPI), and ERC20 token contracts. These dependencies significantly impact the overall security and functionality of the contracts.

OpenZeppelin Libraries: Smart contracts utilize *OpenZeppelin* Ownable and ERC20 libraries. These libraries are well-tested and widely used in the Ethereum community, which adds a layer of trust regarding their implementation. However, they are treated as black boxes in this audit, with an assumption of their functional correctness.

External ERC20 Tokens: Smart contracts interact with external ERC20 tokens for functions like staking, withdrawals, and reward distributions. Any vulnerabilities or changes in these token contracts, such as changes in token logic or decimal precision, could impact the functions of the Ktv2 contract.

Token Price Interface (TPI): Smart contract relies on an external TPI for fetching token prices, which is crucial for financial calculations within the contract. The integrity and availability of this price data are vital for the contract's proper function.

Any interaction or integration with DEX routers or other DeFi platforms introduces a risk if these platforms are compromised or if their APIs change. This can affect the contract's ability to execute token trades or fetch reliable market data.

RECOMMENDATION

Inspect third party dependencies regularly, and mitigate severe impacts whenever necessary.



ACKNOWLEDGEMENT

Shina Inu team is committed to conducting regular inspections of all dependencies and will implement updates as necessary to ensure optimal performance and security.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL



Identifier	Definition	Severity
COD-12	Lack of event-driven architecture	Minor ●


Smart contract uses function calls to update state, which can make it difficult to track and analyze changes to the contract over time.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

RECOMMENDATION

Use events to track state changes. Events improve transparency and provide a more granular view of contract activity.



Identifier	Definition	Severity
VOL-01	Irrelevant code	Minor 

Redundant code found in:

IERC20.sol

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

RECOMMENDATION

Remove redundant and dead code.

RESOLUTION

Shina Inu team argued that IERC20 code does not implement/influence existing logic and has decided to maintain the current implementation without changes.



DISCLAIMERS

InterFi Network provides the easy-to-understand audit of solidity source codes (commonly known as smart contracts).

The smart contract for this particular audit was analyzed for common contract vulnerabilities, and centralization exploits. This audit report makes no statements or warranties on the security of the code. This audit report does not provide any warranty or guarantee regarding the absolute bug-free nature of the smart contract analyzed, nor do they provide any indication of the client's business, business model or legal compliance. This audit report does not extend to the compiler layer, any other areas beyond the programming language, or other programming aspects that could present security risks. Cryptographic tokens are emergent technologies, they carry high levels of technical risks and uncertainty. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. This audit report could include false positives, false negatives, and other unpredictable results.

CONFIDENTIALITY

This report is subject to the terms and conditions (including without limitations, description of services, confidentiality, disclaimer and limitation of liability) outlined in the scope of the audit provided to the client. This report should not be transmitted, disclosed, referred to, or relied upon by any individual for any purpose without InterFi Network's prior written consent.

NO FINANCIAL ADVICE

This audit report does not indicate the endorsement of any particular project or team, nor guarantees its security. No third party should rely on the reports in any way, including to make any decisions to buy or sell a product, service or any other asset. The information provided in 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. This audit report should not be used in any way



to make decisions around investment or involvement. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort.

FOR AVOIDANCE OF DOUBT, SERVICES, INCLUDING ANY ASSOCIATED AUDIT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

TECHNICAL DISCLAIMER

ALL SERVICES, AUDIT REPORTS, SMART CONTRACT AUDITS, OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, INTERFI NETWORK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO SERVICES, AUDIT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, INTERFI NETWORK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM THE COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

WITHOUT LIMITING THE FOREGOING, INTERFI NETWORK MAKES NO WARRANTY OF ANY KIND THAT ALL SERVICES, AUDIT REPORTS, SMART CONTRACT AUDITS, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET THE CLIENT'S OR ANY OTHER INDIVIDUAL'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE.

TIMELINESS OF CONTENT

The content contained in this audit report is subject to change without any prior notice. InterFi Network does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following the publication.



LINKS TO OTHER WEBSITES

This audit report provides, through hypertext or other computer links, access to websites and social accounts operated by individuals other than InterFi Network. Such hyperlinks are provided for your reference and convenience only and are the exclusive responsibility of such websites' and social accounts' owners. You agree that InterFi Network is not responsible for the content or operation of such websites and social accounts and that InterFi Network shall have no liability to you or any other person or entity for the use of third-party websites and social accounts. You are solely responsible for determining the extent to which you may use any content at any other websites and social accounts to which you link from the report.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL



ABOUT INTERFI NETWORK

InterFi Network provides intelligent blockchain solutions. We provide solidity development, testing, and auditing services. We have developed 150+ solidity codes, audited 1000+ smart contracts, and analyzed 500,000+ code lines. We have worked on major public blockchains e.g., Ethereum, Binance, Cronos, Doge, Polygon, Avalanche, Metis, Fantom, Bitcoin Cash, Velas, Oasis, etc.

InterFi Network is built by engineers, developers, UI experts, and blockchain enthusiasts. Our team currently consists of 4 core members, and 6+ casual contributors.

Website: <https://interfi.network>

Email: hello@interfi.network

GitHub: <https://github.com/interfinetwork>


Telegram (Engineering): <https://t.me/interfiaudits>

Telegram (Onboarding): <https://t.me/interfisupport>



 interfinetwork

 hello@interfi.network

 <https://interfi.network>

SMART CONTRACT AUDITS | SOLIDITY DEVELOPMENT AND TESTING
RELENTLESSLY SECURING PUBLIC AND PRIVATE BLOCKCHAINS