



SMART CONTRACT AUDIT

 interfinetwork

 hello@interfi.network

 <https://interfi.network>

PREPARED FOR

OMNI MINER



INTRODUCTION

Auditing Firm	InterFi Network
Client Firm	Omni Miner
Methodology	Automated Analysis, Manual Code Review
Language	Solidity
Contract	0x18ca6173b8adb046cad63efa63d32037270bd464
Blockchain	Sonic
Centralization	Active Ownership
Commit	d25fdcf7be70458019931ca78a4abb190e2b88a1
Website	https://omniminer.io/
Report Date	March 30, 2025


 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	3	1	7	0
Acknowledged	0	1	1	0	1
Resolved	0	0	0	0	0
Important Functions	hatchEggs(), sellEggs(), buyEggs()				
Important Privileges	seedMarket()				

 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.


 Please note that the absence of public KYC verification of the project owners, team members, or deployers associated with Omni Miner. Typically, third-party KYC processes are instrumental in ensuring the transparency and accountability of a project's leadership, thereby enhancing user trust and regulatory compliance. Without external KYC verification by reputable providers, users may face increased risks related to rug pull.



TABLE OF CONTENTS

TABLE OF CONTENTS 3

SCOPE OF WORK..... 5

AUDIT METHODOLOGY 6

RISK CATEGORIES 8

CENTRALIZED PRIVILEGES 9

AUTOMATED ANALYSIS..... 10

INHERITANCE GRAPH12

MANUAL REVIEW13

DISCLAIMERS 29

ABOUT INTERFI NETWORK 32

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



SCOPE OF WORK

InterFi was consulted by Omni Miner to conduct the smart contract audit of their solidity source codes.

The audit scope of work is strictly limited to mentioned solidity file(s) only:

- omniminer.sol

i 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://sonicscan.org/address/0x18ca6173b8adb046cad63efa63d32037270bd464#code	
Contract Name	omniminer
Compiler Version	0.8.9
License	MIT



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

- 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

```

| **SafeMath** | Library |   | |
| L | tryAdd | Internal  |   |
| L | trySub | Internal  |   |
| L | tryMul | Internal  |   |
| L | tryDiv | Internal  |   |
| L | tryMod | Internal  |   |
| L | add | Internal  |   |
| L | sub | Internal  |   |
| L | mul | Internal  |   |
| L | div | Internal  |   |
| L | mod | Internal  |   |
| L | sub | Internal  |   |
| L | div | Internal  |   |
| L | mod | Internal  |   |
|||||
| **Context** | Implementation |   |
| L | _msgSender | Internal  |   |
| L | _msgData | Internal  |   |

```



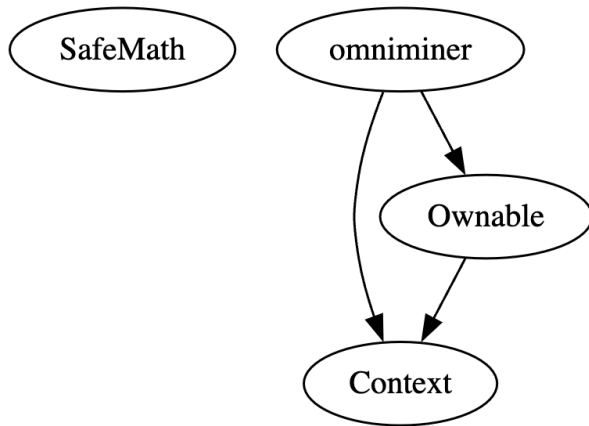
|||||

| ****Ownable**** | Implementation | Context |||| ^L | <Constructor> | Public ! | 🚫 | NO ! || ^L | owner | Public ! | | NO ! || ^L | `renounceOwnership` | Public ! | 🚫 | onlyOwner || ^L | `transferOwnership` | Public ! | 🚫 | onlyOwner || ^L | `_transferOwnership` | Internal 🔒 | 🚫 | |

|||||

| ****omniminer**** | Implementation | Context, Ownable |||| ^L | <Constructor> | Public ! | 🚫 | NO ! || ^L | `hatchEggs` | Public ! | 🚫 | NO ! || ^L | `sellEggs` | Public ! | 🚫 | NO ! || ^L | `beanRewards` | Public ! | | NO ! || ^L | `buyEggs` | Public ! | 💰 | NO ! || ^L | `calculateTrade` | Private 🔒 | | || ^L | `calculateEggSell` | Public ! | | NO ! || ^L | `calculateEggBuy` | Public ! | | NO ! || ^L | `calculateEggBuySimple` | Public ! | | NO ! || ^L | `devFee` | Private 🔒 | | || ^L | `seedMarket` | Public ! | 💰 | onlyOwner || ^L | `getBalance` | Public ! | | NO ! || ^L | `getMyMiners` | Public ! | | NO ! || ^L | `getMyEggs` | Public ! | | NO ! || ^L | `getEggsSinceLastHatch` | Public ! | | NO ! || ^L | `min` | Private 🔒 | | |TERFI
CONFIDENTIALINTERFI
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 🟡

Important onlYowner centralized privileges are listed below:

```
renounceOwnership()  
transferOwnership()  
seedMarket()
```

RECOMMENDATION

Securing private keys or access credentials of deployers, contract owners, operators, 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

Omni Miner team argued that centralized and controlled privileges are used as required.



Identifier	Definition	Severity
CEN-02	First Depositor Advantage (Market Initialization Vulnerability)	Major 🟡

Function `seedMarket()` initializes the `marketEggs` to a fixed value (108000000000) and sets `initialized = true`. However, no ETH is required to seed the contract, and the price calculation in `calculateTrade()` (used by `calculateEggBuy()`) heavily favors the first buyer when the contract's ETH balance is near zero.

In `calculateTrade()`:

```
return (PSN * bs) / (PSNH + ((PSN * rs + PSNH * rt) / rt));
```

When `bs` is near zero and `rt` is very small, the formula returns an extremely large number of eggs to the buyer — giving the first buyer a massive advantage in the number of miners they can generate.

RECOMMENDATION

`seedMarket()` should be modified to accept an initial BNB deposit along with setting the initial `marketEggs`. This deposit should be substantial enough to mitigate the first depositor advantage. The best practice is to seed the market with an amount of BNB and eggs that reflects a reasonable starting price.



Identifier	Definition	
CEN-04	Lack of proxy and upgradeable contracts	

Privileged role cannot authorize contract upgrade. Contract upgradeability allows privileged roles to change current contract implementation in case of a hack.

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

RECOMMENDATION

Employ proxy mechanism to enable future upgradeability. This design allows for seamless upgrades to contract functionality, ensuring better contract maintainability and adaptability.



Identifier	Definition	Severity
LOG-02	Potential front-running	Medium 🟡

Potential front-running happens when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by front-running a transaction to purchase assets and make profits by back-running a transaction to sell assets. Below mentioned functions are called without setting restrictions on slippage or minimum output:

hatchEggs

sellEggs

buyEggs

TERFI
CONFIDENTIAL

INTERFI
CONFIDENTIAL

RECOMMENDATION

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

ACKNOWLEDGEMENT

According to OmniMiner team, front-running is not a concern in this contract due to its specific design and mechanics. Transactions adhere to a predefined contract logic, which ensures that users cannot manipulate the execution order to gain an unfair advantage. Additionally, egg-related transactions—such as selling or hatching—are tied to an individual user's state, meaning these actions are personal and remain unaffected by other users attempting to jump ahead in the transaction queue. Smart contract also prevents users from directly selling their full position, adding another layer of protection against potential front-running exploits.



Identifier	Definition	Severity
LOG-03	Re-entrancy	Major 🟡

Below mentioned function does not follow Checks-Effects-Interactions (CEI) pattern when transferring control to external entities:

`sellEggs`

- Function sends ETH to the user (`payable(msg.sender).transfer(...)`) before updating internal state (`claimedEggs`, `lastHatch`, etc).
- Although `transfer()` limits gas, future versions of Solidity or certain proxies/wrappers may re-enable risk.

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

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.



Identifier	Definition	Severity
COD-02	Timestamp dependence	Minor ●

Be aware that the timestamp of the block can be manipulated by miners. Since miners can slightly adjust the timestamp, they may influence contract outcomes to their advantage.

Timestamp Dependency in `getEggsSinceLastHatch()`

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

RECOMMENDATION

Avoid relying solely on timestamp of the block for critical contract functions. Follow 15 seconds rule, and scale time dependent events accordingly.



Identifier	Definition	Severity
COD-03	No limit on egg inflation	Major 🟡


- There's no cap on how many eggs users can accumulate or how many miners can be generated.
- `marketEggs` is adjusted arbitrarily in `hatchEggs()` and `sellEggs()`, making it manipulable.

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

RECOMMENDATION

Cap maximum number of miners or eggs per address, and implement fixed rate exchange or auction-based pricing.



Identifier	Definition	Severity
COD-06	Hardcoded accounts	Minor 

- recAdd is hardcoded to deployer. If compromised, fees could be redirected.
- devFeeVal is hardcoded but not updatable — no way to change it in emergencies or stop abusive settings.

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

RECOMMENDATION

Private keys of externally owned accounts must be secured carefully.



Identifier	Definition	Severity
COD-07	Arbitrary external calls	Medium 🟡

Smart contract functions use `.transfer()`, but in newer versions, the gas stipend may not be enough for contracts to receive ETH. This can break maintainability and integration with wallets, proxies, or smart contract wallets.

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

RECOMMENDATION

Provide pull-based withdrawals (safer architecture) or use `.call()` with success check.



Identifier	Definition
COD-08	Lack of fallback function

Fallback functions are usually executed in one of the following cases: If a function identifier doesn't match any of the available functions in a smart contract. If there was no data supplied along with the function call. Use fallback function with empty data, and mark it external, and payable.

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-10	Direct and indirect dependencies	Unknown 🟤

Scope of this audit treats Ownable and Context contracts as black boxes and assumes their functional correctness. The security and stability of the Sonic Chain are also outside the scope of this audit. We assume centralized addresses are securely managed. While this contract avoids direct interaction with complex external DeFi protocols, the broader ecosystem risks associated with blockchain technology and cryptocurrency still apply.


RECOMMENDATION

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

ACKNOWLEDGEMENT

OmniMiner team will inspect third party dependencies regularly, and push upgrades whenever required.



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.

```
buyEggs()  
sellEggs()  
hatchEggs()
```

RECOMMENDATION

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



Identifier	Definition	Severity
COM-02	Outdated SafeMath in Solidity	Minor 

SafeMath in Solidity 0.8.9, already has built-in overflow/underflow protection. This makes SafeMath redundant and bloats bytecode.

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

RECOMMENDATION

Remove all instances of SafeMath and use native operators.



Identifier	Definition	Severity
COM-03	Poor naming	Minor ●


- Functions like `beanRewards()` use “beans” while the rest use “eggs”.
- Variable names like `EGGS_T0_HATCH_1MINERS` are descriptive, but not PSN, PSNH.

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

RECOMMENDATION

Standardize naming for readability and maintainability.



Identifier	Definition	Severity
COM-04	Unbounded loop risk	Minor 

Smart contract doesn't prevent referral tree loops or spamming multiple self-referrals:


```
event ReferralSet(address user, address referrer)
```

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

RECOMMENDATION

Add logic to prevent self-referrals and limit referral depth (1 or 2).



Identifier	Definition	Severity
COM-05	Potential griefing (DoS via referral spamming)	Minor 

A user could spam `hatchEggs()` with fake refs, bloating storage.

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

RECOMMENDATION

Use stricter validation and storage cleanup.



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