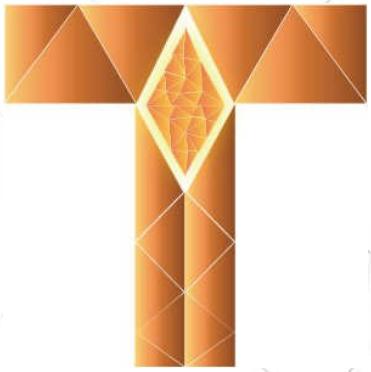
InterFi NETWORK

@INTERFINETWORK

SMART CONTRACT SECURITY AUDIT

Tsar Network



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
Project Overview Approved BY	Albert | Project Specialist at InterFi Network
Platform	Solidity
Audit Check (Mandatory)	Static & Manual Analysis
Project Check (Optional)	KYC Analysis
Consultation Request Date	September 29, 2021
Report Date	October 03, 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:

- **Tsar Network's smart contract source code has LOW RISK SEVERITY.**
- **Tsar Network has successfully PASSED the smart contract audit.**
- **Tsar Network has successfully PASSED the owner KYC verification.**

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

Table Of Contents

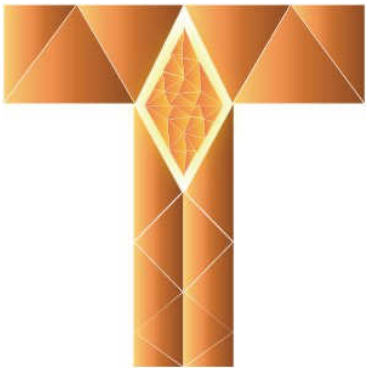
Project Overview

InterFi was consulted by Tsar Network on September 29, 2021 to conduct a smart contract security audit of their token source code.

Tsar wants to create equilibrium by providing a platform that is both dev-friendly as well as user-friendly. Blockchain cannot get mainstream until technical barriers are broken and the non-codeholic majority is taken into consideration. Tsar zero-code approach lets non-technical users focus on the business, not the code.

Project	TSAR NETWORK
Blockchain	Binance Smart Chain (Proprietary Tsar Chain In Development)
Language	Solidity
Contract	0x2b697fe168fb168e34153b775c5fd7cbcaa30b75
Website	https://tsar.network/en/
Whitepaper	https://filedn.com/lNp0q9H6YW0RRflpVmxyi6m/TsarNetwork/whitepaper.pdf
Telegram	https://t.me/TsarNetwork
Telegram Channel	https://t.me/TsarNetworkChannel
Twitter	https://twitter.com/tsarnetwork
Reddit	https://www.reddit.com/r/TsarNetwork/
Facebook	https://www.facebook.com/TsarNetwork-105481808546769

Public logo



Solidity Source Code On Smart Chain (BscScan Verified Source Code)

https://bscscan.com/token/0x2b697fe168fb168e34153b775c5fd7cbcaa30b75

Contract Name: TSAR

Compiler Version: v0.8.7+commit.e28d00a7

Optimization Enabled: Yes with 200 runs

Solidity Source Code On InterFi GitHub

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

GitHub Commits

Solidity source code committed at: 3b713e1e4f2d70be4d5d377958d3ac105c4d9a99

Audit Scope & Methodology

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

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

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
Source Code Review	❖ Gas Limit and Loops ❖ 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

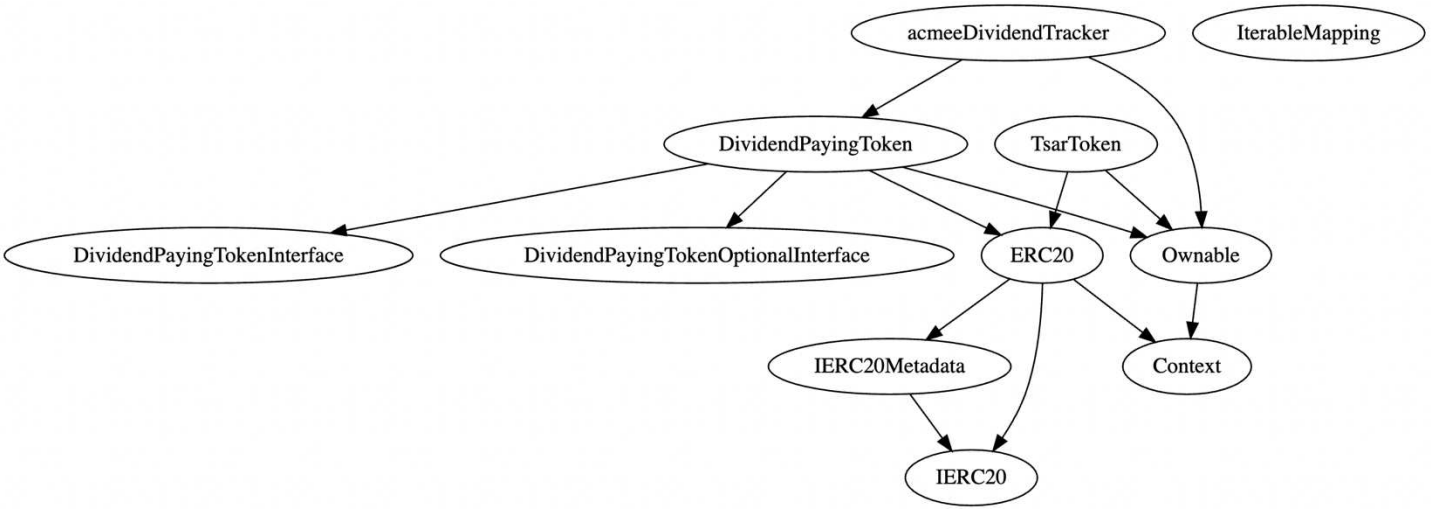
Symbol	Meaning
🛑	Function can be modified
💵	Function is payable
🔒	Function is locked
🔐	Function can be accessed
!	Important functionality

Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
DividendPayingTokenInterface	Interface			
└	dividendOf	External !		NO !
└	withdrawDividend	External !	🛑	NO !
DividendPayingTokenOptionalInterface	Interface			
└	withdrawableDividendOf	External !		NO !
└	withdrawnDividendOf	External !		NO !
└	accumulativeDividendOf	External !		NO !
IERC20	Interface			
└	totalSupply	External !		NO !
└	balanceOf	External !		NO !
└	transfer	External !	🛑	NO !
└	allowance	External !		NO !
└	approve	External !	🛑	NO !
└	transferFrom	External !	🛑	NO !
IERC20Metadata	Interface	IERC20		
└	name	External !		NO !
└	symbol	External !		NO !
└	decimals	External !		NO !
IterableMapping	Library			
└	get	Public !		NO !
└	getIndexOfKey	Public !		NO !
└	getKeyAtIndex	Public !		NO !
└	size	Public !		NO !

```
| └ | set | Public ❗ | 🛑 |NO❗ |
| └ | remove | Public ❗ | 🛑 |NO❗ |
||||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata |||
| └ | <Constructor> | Public ❗ | 🛑 |NO❗ |
| └ | name | Public ❗ | |NO❗ |
| └ | symbol | Public ❗ | |NO❗ |
| └ | decimals | Public ❗ | |NO❗ |
| └ | totalSupply | Public ❗ | |NO❗ |
| └ | balanceOf | Public ❗ | |NO❗ |
| └ | transfer | Public ❗ | 🛑 |NO❗ |
| └ | allowance | Public ❗ | |NO❗ |
| └ | approve | Public ❗ | 🛑 |NO❗ |
| └ | transferFrom | Public ❗ | 🛑 |NO❗ |
| └ | increaseAllowance | Public ❗ | 🛑 |NO❗ |
| └ | decreaseAllowance | Public ❗ | 🛑 |NO❗ |
| └ | _transfer | Internal 🔒 | 🛑 | |
| └ | _mint | Internal 🔒 | 🛑 | |
| └ | _burn | Internal 🔒 | 🛑 | |
| └ | _approve | Internal 🔒 | 🛑 | |
| └ | _beforeTokenTransfer | Internal 🔒 | 🛑 | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑 |NO❗ |
| └ | owner | Public ❗ | |NO❗ |
| └ | renounceOwnership | Public ❗ | 🛑 | onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑 | onlyOwner |
||||||
| **SafeMath** | Library | |||
| └ | add | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | mul | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | mod | Internal 🔒 | | |
| └ | mod | Internal 🔒 | | |
||||||
| **SafeMathInt** | Library | |||
| └ | mul | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | add | Internal 🔒 | | |
| └ | abs | Internal 🔒 | | |
| └ | toUint256Safe | Internal 🔒 | | |
||||||
| **SafeMathUint** | Library | |||
| └ | toInt256Safe | Internal 🔒 | | |
||||||
||||||
```

TsarToken	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	buyFee	Internal 🔒	🛑	
└	sellFee	Internal 🔒	🛑	
└	updateDividendTracker	Public ❗	🛑	onlyOwner
└	updateUniswapV2Router	Public ❗	🛑	onlyOwner
└	excludeFromFees	Public ❗	🛑	onlyOwner
└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner
└	setMarketingWallet	External ❗	🛑	onlyOwner
└	setacmeeRewardsFee	External ❗	🛑	onlyOwner
└	setLiquiditFee	External ❗	🛑	onlyOwner
└	setMarketingFee	External ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	blacklistAddress	External ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	isExcludedFromFees	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗
└	processDividendTracker	External ❗	🛑	NO❗
└	claim	External ❗	🛑	NO❗
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfDividendTokenHolders	External ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	swapAndSendToFee	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	
└	swapTokensForEth	Private 🔐	🛑	
└	swapTokensForBNB	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	swapAndSendDividends	Private 🔐	🛑	

Callout functions – Inheritance Graph



Smart Contract – Manual Analysis

1. **Tsar Network smart contract tax & wallet data.**

```
contract TSAR is ERC20, Ownable {
    using SafeMath for uint256;

    bool private swapping;
    TsarDividendTracker public dividendTracker;
    address public deadWallet = 0x000000000000000000000000000000000000dEaD;
    address public immutable BNB = address(0xB8c77482e45F1F44dE1745F52C74426C631bDD52);
    uint256 public swapTokensAtAmount = 20000000 * (10**18);
    mapping(address => bool) public _isBlacklisted;
    uint256 public BNBRewardsFee;
    uint256 public liquidityFee;
    uint256 public marketingFee;
    uint256 public fireWhaleFee;
    uint256 public totalFees;
    address public _marketingWalletAddress = 0x280ee73d592d9A3Da6786903e366634cfb02Ed74;
    address public _fireWhaleAddress  = 0x444e7816324cdb98AD6398fF37343598bf6a9C83;
    uint256 public maxTxAmount = 10000000000 * (10**18)
```

2. **Tsar Network Smart Contract utilizes "SafeMath" function to avoid common smart contract vulnerabilities.**


```
library SafeMath {
function add(uint256 a, uint256 b) internal pure returns (uint256) {
    uint256 c = a + b;
    require(c >= a, 'SafeMath: addition overflow');

    return c;
  }
  function sub(uint256 a, uint256 b) internal pure returns (uint256) {
    return sub(a, b, 'SafeMath: subtraction overflow');
  }
  uint256 c = a * b;
    require(c / a == b, 'SafeMath: multiplication overflow');

    return c;
  }
```

3. **Tsar Network smart contract has 1 low severity issue which may or may not create any functional vulnerability.**

   Expected pragma, import directive or contract/interface/library definition.

{

"resource": " /TsarTest.sol",

"owner": "_generated_diagnostic_collection_name_#0",

"severity": 8, (! Low Severity)

"message": "Expected pragma, import directive or contract/interface/library definition.",

"source": "solc",

"startLineNumber": 25,

"startColumn": 1,

"endLineNumber": 25,

"endColumn": 2

}

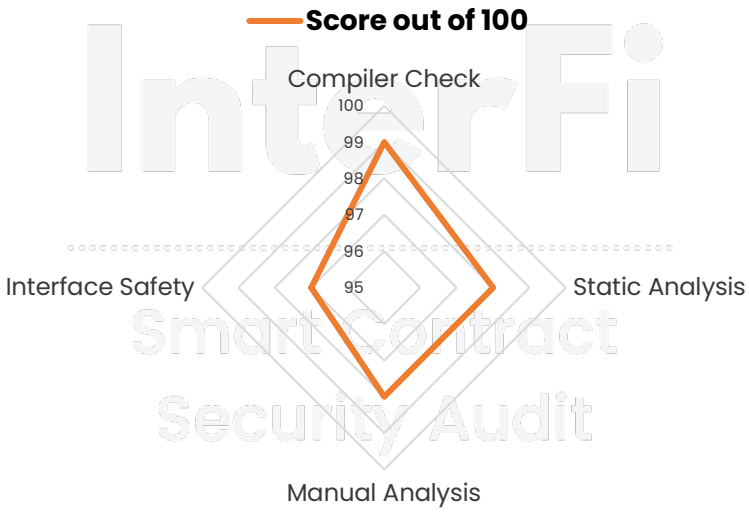
4. **When the smart contract has an active owner address, some of the smart contract functions can be edited, modified or altered.**

5. **Owner: 0xd888817f5f0e047fe453b5205cbccadf5c131fb7**

6. **Total Supply: 10 Billion TSAR**

Smart Contract - Risk Status & Radar Chart

Risk Severity	Status
! Critical	None critical severity issues identified
! High	None high severity issues identified
! Medium	None low severity issues identified
! Low	**1 Low severity issue identified**



Compiler Check	99
Static Analysis	98
Manual Analysis	98
Interface Safety	97

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.

Tsar Network's smart contract source code has LOW RISK SEVERITY.

Tsar Network has successfully PASSED the smart contract audit.

Tsar Network has successfully PASSED the owner KYC verification.

General 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