VIKING SHIELD

*Bring trust into your project*

**Blockchain Security I Smart Contract Audits**

MADE IN SWITZERLAND

# Audit

## Security Assessment

## 22.NOV,2021

For

# contents

# Disclaimer

Viking Shield reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. Viking Shield do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc’...)

**Viking Shield Audits do not provide any warranty or guarantee regarding the absolute bug– free nature of the technology analyzed, nor do they provide any indication of the technology proprietors.**
**Viking Shield Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.**

Viking Shield Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. Viking Shield ’s position is that each company and individual are responsible for their own due diligence and continuous security. Viking Shield in no way claims any guarantee of security or functionality of the technology we agree to analyze.

| Version | Date | Description |
| --- | --- | --- |
| 1.0 | 22.NOV 2021 | Layout project<br>Automated– /Manual–<br>Security Testing<br>Summary |

## Community:

Official English Telegram:

https://t.me/TOTSGAMES_COMMUNITY

Official English Channel:

https://t.me/totsgames_EN

Twitter:

https://twitter.com/TOTSGAMES

## Description

Tower of the sorcerer–a defense game based on Binance Smart Chain.
In this fun tower defense strategy game, a player should defend the tower of the sorcerer from greedy adventurers.  Assemble an army of defenders and fight off greedy adventurers hordes from tower of the sorcerer! Clash with the enemy and make sure that your tower of the sorcerer will hold the line in battle.These battles will leave you unforgettable impressions. The Play to earn feature will provide you with an engaging experience.

Experience and earn real money!

## Tower of the sorcerer NFT ecosystem

Tower of the sorcerer is a comprehensive NFT ecosystem ,  including Tower of the sorcerer system BSC,high lucrative fantasy–themed RPG games with high profit,high liquidity NFT market,  world–class collectibles and Tower of the sorcerer community. This ecosystem is a perfect combination of NFT games and DeFi ,  allowing users to have fun and earn a great deal of money at the same time.

## Project Engagement

During the Date ,  **Tots games Team** engaged Viking Shield to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Viking Shield with access to their code repository and whitepaper.

## Logo


## Contract Link

v1.0

https://bscscan.com/token/0x720ddb3c219c01fd234d37fda1ea09102b97e596

## Vulnerability & Risk Level

Risk represents the probability that a certain source–threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

| Level | Value | Vulnerability | Risk (Required Action) |
|---|---|---|---|
| **Critical** | 9 – 10 | A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken. | Immediate action to reduce risk level. |
| **High** | 7– 8.9 | A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way. | Implementation of corrective actions as soon as possible. |
| **Medium** | 4– 6.9 | A vulnerability that could affect the desired outcome of executing the contract in a specific scenario. | Implementation of corrective actions in a certain period. |
| **Low** | 2– 3.9 | A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective. | Implementation of certain corrective actions or accepting the risk. |

| | | | |
|---|---|---|---|
| **Informational** | 0 — 1.9 | A vulnerability that have informational character but is not effecting any of the code. | An observation that does not determine a level of risk |

## Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

## Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
    i) Review of the specifications, sources, and instructions provided to Viking Shield to make sure we understand the size, scope, and functionality of the smart contract.
    ii) Manual review of code, which is the process of reading source code line-by- line in an attempt to identify potential vulnerabilities.
    iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Viking Shield describe.

2. Testing and automated analysis that includes the following:
    i) Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    ii) Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.

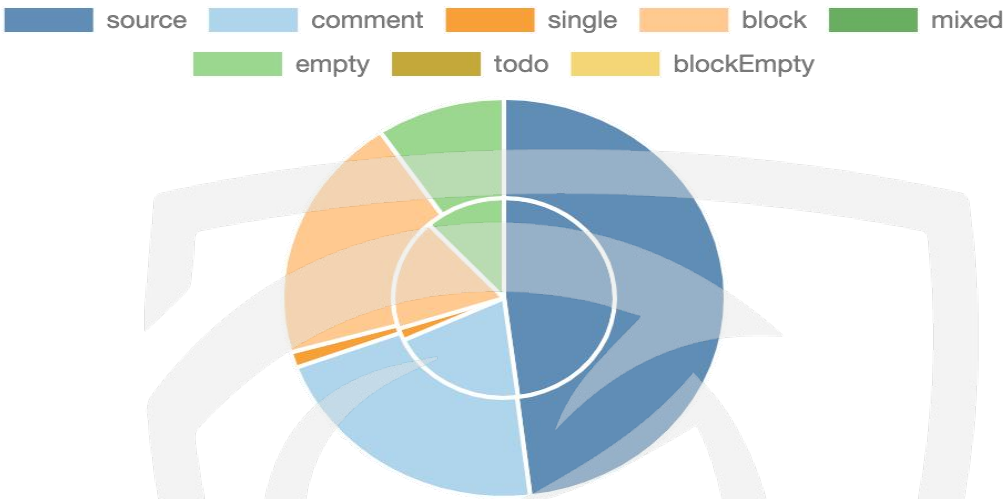
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

# Metrics

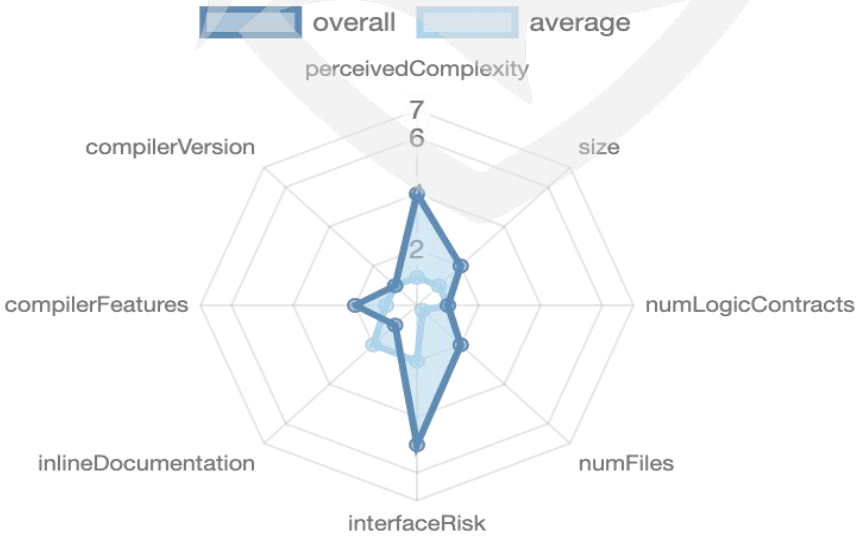
## Source Lines

v1.0




## Risk Level

v1.0



# Capabilities

## Components

| Version | Contracts | Libraries | Interfaces | Abstract |
|---|---|---|---|---|
| 1.0 | 1 | 1 | 4 | 2 |

## Exposed Functions

*This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.*

| Version | Public | Payable |
|---|---|---|
| 1.0 | 63 | 4 |

| Version | External | Internal | Private | Pure | View |
|---|---|---|---|---|---|
| 1.0 | 47 | 64 | 3 | 11 | 19 |

## State Variables

| Version | Total | Public |
|---|---|---|
| 1.0 | 17 | 6 |

## Capabilities

| Version | Solidity Versions observed | Experimental Features | Can Receive Funds | Uses Assembly | Has Destroyable Contracts |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| 1.0 | `^0.8.0` | `ABIEncoderV2` | `yes` | `yes` (2 asm blocks) | |

| Version | Transfers ETH | Low-Level Calls | DelegateCall | Uses Hash Functions | ECRecover | New/Create/Create2 |
|---|---|---|---|---|---|---|
| 1.0 | `yes` | | `yes` | | | |

## Scope of Work

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

1.Correct implementation of Token standard

2.Deployer cannot mint any new tokens

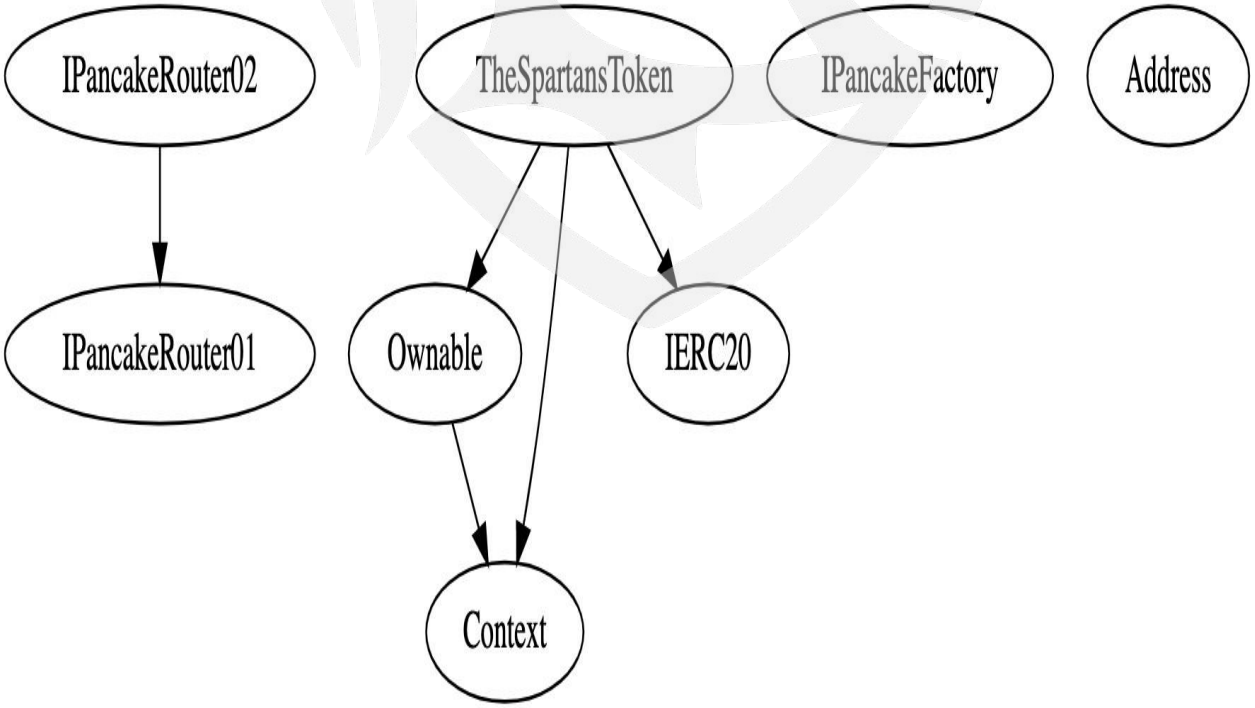
3.Deployer cannot burn or lock user funds

4.Deployer cannot pause the contract

5.Overall checkup (Smart Contract Security)

## Inheritance Graph

v1.0



# Verify Claims

## Correct implementation of Token standard

| Tested | Verified |
| --- | --- |
| ✓ | ✓ |

| Function | Description | Exist | Tested | Verified |
| --- | --- | --- | --- | --- |
| TotalSupply | provides information about the total token supply | ✓ | ✓ | ✓ |
| BalanceOf | provides account balance of the owner's account | ✓ | ✓ | ✓ |
| Transfer | executes transfers of a specified number of tokens to a specified address | ✓ | ✓ | ✓ |
| TransferFrom | executes transfers of a specified number of tokens from a specified address | ✓ | ✓ | ✓ |
| Approve | allow a spender to withdraw a set number of tokens from a specified account | ✓ | ✓ | ✓ |
| Allowance | returns a set number of tokens from a spender to the owner | ✓ | ✓ | ✓ |

## Optional implementations

| Function | Description | Exist | Tested | Verified |
| --- | --- | --- | --- | --- |
| renounceOwnership | Owner renounce ownership for more trust | ✓ | ✓ | ✓ |

## Deployer cannot mint any new tokens

| Name | Exist | Tested | Verified | File |
|---|---|---|---|---|
| Deployer cannot mint | ✓ | ✓ | ✓ | Main |
| Comment | Line: – | | | |

Max / Total Supply: 200.000.000

## Deployer cannot burn or lock user funds

| Name | Exist | Tested | Verified |
|---|---|---|---|
| Deployer cannot lock | ✓ | ✓ | ✓ |
| Deployer cannot burn | ✓ | ✓ | ✓ |

## Deployer cannot pause the contract

| Name | Exist | Tested | Verified |
|---|---|---|---|
| Deployer cannot pause | ✓ | ✓ | ✓ |

## Overall checkup (Smart Contract Security)

| Tested | Verified |
| --- | --- |
| ✓ | ✓ |

### Legend

| Attribute | Symbol |
| --- | --- |
| Verfified / Checked | ✓ |
| Partly Verified | ⚑ |
| Unverified / Not checked | ✗ |
| Not available | – |

# Source Units in Scope

## v1.0

### Legend

| Attribute | Description |
| --- | --- |
| Lines | total lines of the source unit |
| nLines | normalized lines of the source unit (e.g. normalizes functions spanning multiple lines) |
| nSLOC | normalized source lines of code (only source-code lines; no comments, no blank lines) |
| Comment Lines | lines containing single or block comments |
| Complexity Score | a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...) |

# Audit Results

## AUDIT PASSED

## Critical issues

- no critical issues found -

## High issues

- no high issues found -

## Medium issues

- no medium issues found -

## Low issues

| Issue | File | Type | Line | Description |
|---|---|---|---|---|
| #1 | Main | A floating pragma is set | 2 | The current pragma Solidity directive is „“^0.8.0““. |
| #2 | Main | Missing Zero Address Validation (missing-zero-check) | 48, 42, 233, 229 | Check that the address is not zero |
| #3 | Main | Functions that are not used (dead-code) | 210 | Remove unused functions |

## Informational issues

| Issue | File | Type | Line | Description |
|---|---|---|---|---|
| #1 | Main | State variables that could be declared constant (constable-states) | 32 | Add the `constant` attributes to state variables that never change |

## Audit Comments

22. November 2021:

- Deployer can lock user funds
- Deployer can set address as blacklist, blacklisted addresses (sender/receiver) are not allowed to use transfer function anymore
- Deployer can set trading time to a high value to lock user funds

## Audit details:

| Title | Relationships | Status |
|---|---|---|
| 1 | Unencrypted Private Data On-Chain | passed |
| 2 | Code With No Effects | passed |
| 3 | Message call with hardcoded gas amount | passed |
| 4 | Hash Collisions With Multiple Variable Length Arguments | passed |
| 5 | Unexpected Ether balance | passed |
| 6 | Presence of unused variables | passed |
| 7 | Right-To-Left- Override control character (U+202E) | passed |
| 8 | Typographical Error | passed |
| 9 | DoS With Block Gas Limit | passed |
| 10 | Arbitrary Jump with Function Type Variable | passed |
| 11 | Incorrect Inheritance Order | passed |
| 12 | Write to Arbitrary Storage Location | passed |
| 13 | Requirement Violation | passed |
| 14 | Lack of Proper Signature Verification | passed |
| 15 | Missing Protection against Signature Replay Attacks | passed |
| 16 | Weak Sources of Randomness from Chain Attributes | passed |
| 17 | Shadowing | passed |
| 18 | State Variables | passed |

| | | |
|---|---|---|
| 19 | Incorrect Constructor Name | passed |
| 20 | Signature Malleability | passed |
| 21 | Timestamp Dependence | passed |
| 22 | Authorization through tx.origin | passed |
| 23 | Transaction Order Dependence | passed |
| 24 | DoS with Failed Call | passed |
| 25 | Delegatecall to Untrusted Callee | passed |
| 26 | Use of Deprecated Solidity Functions | passed |
| 27 | Assert Violation | passed |
| 28 | Uninitialized Storage Pointe | passed |
| 29 | State Variable Default Visibility | passed |
| 30 | Reentrancy | passed |
| 31 | Unprotected SELFDESTRUCT Instruction | passed |

# Viking Shield

**Blockchain Security I Smart Contract Audits**

MADE IN SWITZERLAND