

expellee

Building the Futuristic **Blockchain Ecosystem**




Security Audit Report FOR



DeFiTankLand

OVERVIEW

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

 Audit Result	Passed with high risk
 KYC Verification	Yes
 Audit Date	8 Feb 2023

Passed with high risk

-Team Expelee

PROJECT DESCRIPTION

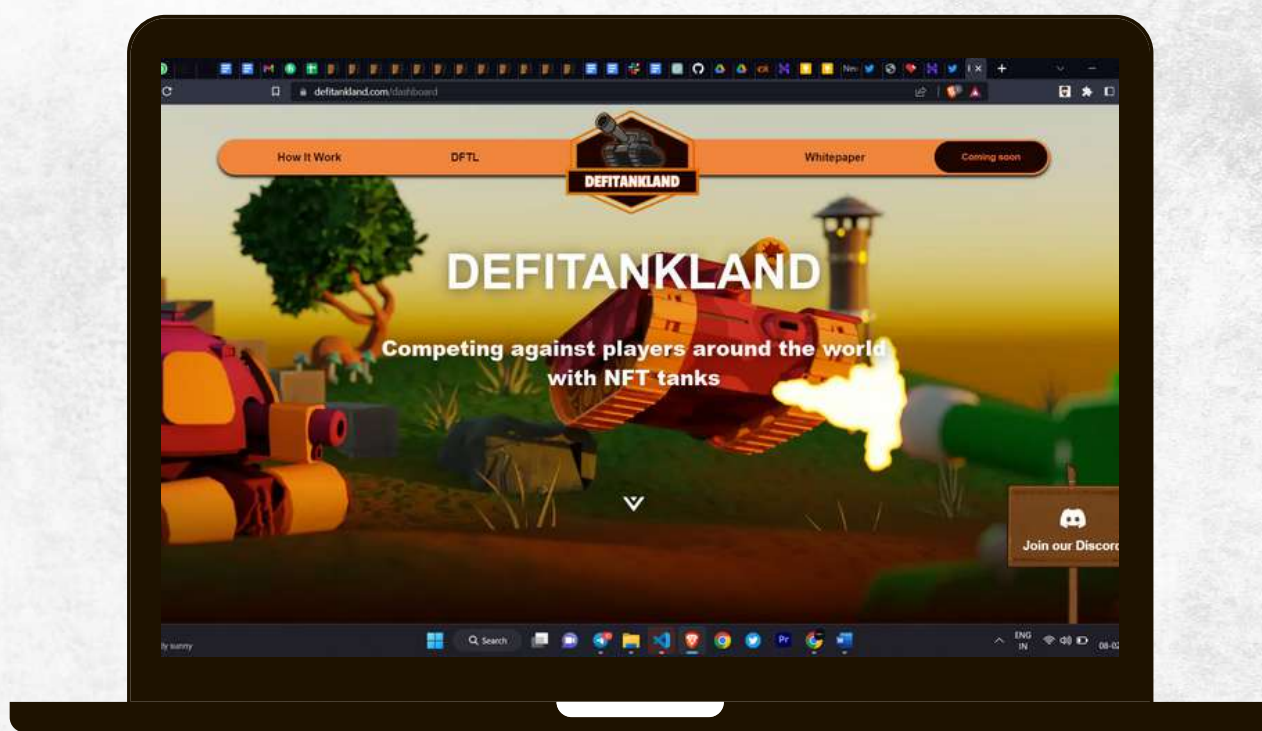
DeFiTankLand

Competing against players around the world with NFT tanks



Social Media Profiles

DeFiTankLand



<https://defitankland.com/dashboard>



<https://twitter.com/defitankland>

It's always good to check the social profiles of the project,
before making your investment.

-Team Expelee

CONTRACT DETAILS

Token Name

DeFiTankLand Token

Symbol

DFTL

Network

Arbitrum

Language

Solidity

Contract Address (Verified)

0xCb6460D56825ddC12229C7a7D94B6b26a9f9C867

Token Type

ERC20

Total Supply

3, 000, 000

Compiler

v0.8.10+commit.fc410830

Optimization Enabled

Yes with 200 runs

Contract SHA-256 Checksum:

c142dd53eb0702282c50bfec22b6148200933a973a92075e0f562535680609ab

Owner's Wallet

0x0000f2f41ee26f355747eda99e760064d9b30a16

Deployer's Wallet

0x0000f2f41ee26f355747eda99e760064d9b30a16

AUDIT METHODOLOGY



Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.



Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.



Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability



Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

FUNCTION OVERVIEW

Can Take Back Ownership

Not Detected

Owner Change Balance

Not Detected

Blacklist

Not Detected

Modify Fees

Not Detected

Proxy

Not Detected

Whitelisted

Not Detected

Anti Whale

Not Detected

Trading Cooldown

Not Detected

Transfer Pausable

Not Detected

Cannot Sell All

Not Detected

Hidden Owner

Not Detected

Mint

Detected

Disable Trades

Not Detected

VULNERABILITY CHECKLIST

Design Logic	Passed
Compiler warnings.	Passed
Private user data leaks	Passed
Timestamp dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious Event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zeppelin module	Passed
Fallback function security	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Low Risk

Issues on this level are minor details and warning that can remain unfixed.

Informational

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.

AUDIT SUMMARY

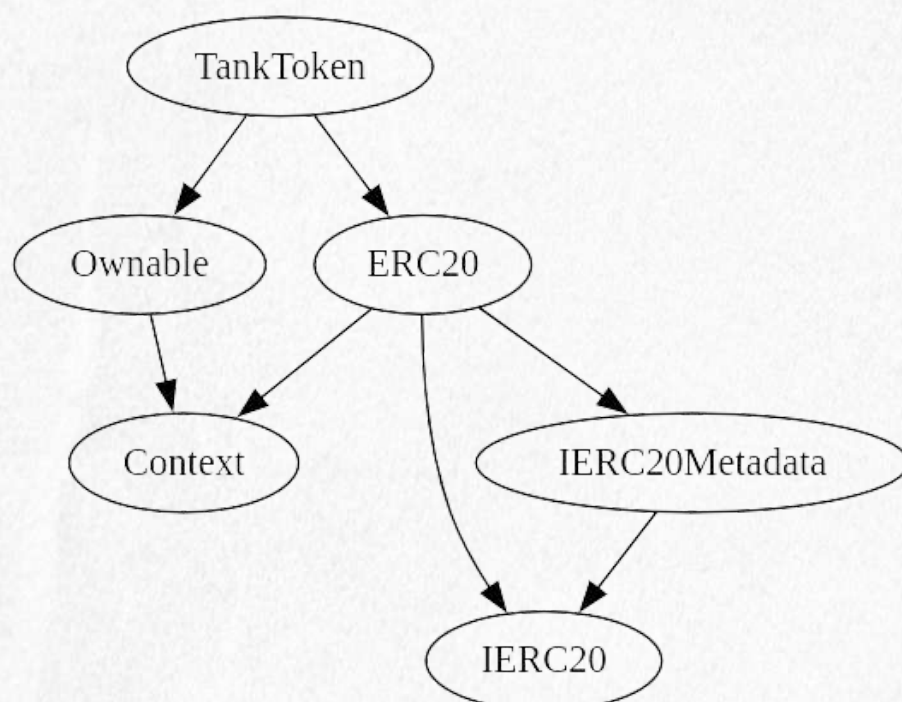
Tools

1- Manual review:

a line by line code review has been performed by expelee team.

2- Slither : static analysis

Inheritance Trees:



Static Analysis

A static analysis of contract's source code has been performed using slither. No major issues were found in the output

```
Context._msgData() (contracts/TestToken.sol#13-15) is never used and should be removed  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
Pragma version^0.8.0 (contracts/TestToken.sol#6) allows old versions  
solc-0.8.17 is not recommended for deployment  
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```


Summary

- Owner is able to mint new tokens
- Owner is not able to set buy/sell taxes (0%)
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to set max buy/sell/transfer amounts
- Owner is not able to disable trades

MANUAL AUDIT

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to a methodology based on OWASP standards.

Vulnerabilities are divided into three primary risk categories: **high**, **medium**, and **low**.

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious Input Handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

FINDINGS

- **High Risk Findings:1**
 - **Medium Risk Findings:0**
 - **Low Risk Findings:0**
 - **Suggestions & discussion: 0**
-

High Risk Findings

Minting: A wallet that has minter role, is able to mint unlimited amount of tokens. Minter role can be given to any wallet by owner.

```
ftrace | funcSig
function mint(address to, uint amount) external onlyMinter {
    mint(to, amount);
}
```

Suggestion :

There are multiple ways to go around this issue:

- 1- give clear explanations for investors and proceed with current implementation
- 2- mint needed supply and renounce ownership, as well as disabling (setting to false) all minter wallets
- 3- renounce ownership without minting or giving minter role to any one
- 4- delete mint functionality and redeploy the contract

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 Start-up. Coping up with numerous solutions for blockchain Security and constructing a Web3 Ecosystem from Deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

 www.expelee.com

 expeleeofficial

 expelee

 Expelee

 expelee

 expelee_official

 expelee-co

expelee

Building the Futuristic **Blockchain Ecosystem**

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always Do your own research and protect yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.