ETHOS
AUDIT

World War Inu

Audit Report

Feb. 28, 2022

Contents

Executive Summary

Audit Details

Project Name	World War Inu
Codebase	https://bscscan.com/address/0x0cde5318c5244ba6aaa4974d74e1125798113b4b#code
Source Code	WorldWarInu.sol
Initial Audit Date	Feb. 28, 2022
Revision Dates	NA
Methodology	Manual

Methodology

This audit's objectives are to evaluate:

- Security-related issues
- Code quality
- Relevant documentation
- Adherence to specifications
- Adherence to best practices

This audit examines the possibility of issues existing along the following vectors (but not limited to):

- Single & Cross-Function Reentrancy
- Front Running (Transaction Order Dependence)
- Timestamp dependence
- Integer Overflow and Underflow
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Number rounding errors
- DoS with (Unexpected) Revert
- DoS with Block Gas Limit
- Insufficient gas griefing
- Forcibly sending native currency
- Logical oversights
- Access control
- Centralization of power
- Logic-Specification Contradiction
- Functionality duplication
- Malicious token minting

The code review conducted for this audit follows the following structure:

1. Review of specifications, documentation to assess smart contract functionality
2. Manual, line-by-line review of code
3. Code's adherence to functionality as presented by documentation
4. Automated tool-driven review of smart contract functionality
5. Assess adherence to best practices
6. Provide actionable recommendations

Contract Details

Contract IDs	**0x0cde5318c5244ba6aaa4974d74e1125798113b4b**
Network	BSC
Language	Solidity
Compiler	v0.8.12+commit.f00d7308
Verification Date	Feb. 22, 2022
Contract Type	BEP-20 Token
Libraries	Open Zeppelin

Token Details

Contract Name	WorldWarInu
Symbol	WWI
Decimals	9
Total Supply	1,000,000,000
Max Tx Amount	5,000,000
Max Wallet Amount	25,000,000
Total Tx Tax %	12
Treasury Reflection %	4
Liquidity Provision %	5
Marketing Reflection %	3
Liquidity Provision Trigger	10,000

Risk Levels

LOW

The issue is informational and does not pose an immediate risk, but is relevant to security best practices.

MEDIUM

The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.

HIGH

The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.

EXTREME

The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.

Result Summary

Ethos' audit of the World War Inu token smart contract has concluded with a **POSITIVE** result.

- World War Inu is a community-led DAO project, mitigating many of the typical Owner Priviledge issues since contract changes should be conducted via governance voting.

- Buy/Sell tax values can be updated by the contract owner, however this is by design according to the DAO's core contributors. As long as changes are implemented by governance voting, this is not an issue.

- All other issues were informational only, and acknowledged by the team. No revision is needed since they do not impact the overall security of the token mechanics.

- While the Buy/Sell taxes can be set to any specified value, the distribution of accumulated reflections is determined by the *Share* values.

- Accumulated reflections are sent distributed to a Marketing wallet, Treasury wallet, and also added LP povision over time.

- While taxes are taken on all buy/sell transactions (unless excluded from fees) reflections to respective wallets are only triggered on Sell transactions. This causes buy transactions to cost significantly less gas than sell transactions. This is not an issue on the Binance Smart Chain.

Issues Reported

Severity	Unresolved	Acknowledged	Resolved
Extreme	0	0	0
High	0	0	0
Medium	0	0	0
Low	0	6	0

Issues Summary

ID	Title	Severity	Status
WWI-0	Owner Privilege	Low	Acknowledged
WWI-1	State variable visibility is not set	Low	Acknowledged
WWI-2	Unhandled return value	Low	Acknowledged
WWI-3	Missing zero-address validation	Low	Acknowledged
WWI-4	Environment & function/variable naming mismatch	Low	Acknowledged
WWI-5	Functions that could be declared external	Low	Acknowledged

Detailed Findings

WWI-0 – Owner Privilege

Severity: Low **Status**: Acknowledged

Description: The "*addLiquidity*" function calls the uniswapV2Router.addLiquidityETH function with the "to" address specified as owner() for acquiring the generated LP tokens from the WWI-BNB pool. The contract also allows the owner to update the buy/sell taxes, max wallet size and transaction sizes.

Risk: Over time the _owner address will accumulate a significant portion of LP tokens. If the _owner is an Externally Owned Account, mishandling of its private key can have devastating consequences to the project as a whole.

Recommendation: We advise the to address of the uniswapV2Router.addLiquidityETH function call to be replaced by the contract itself, i.e. address(this), and to restrict the management of the LP tokens within the scope of the contract's business logic. This will also protect the LP tokens from being stolen if the _owner account is compromised. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or via smart-contract based accounts with enhanced security practices, i.e. Multisig wallets.

Feasible solutions that would also mitigate the potential risk:

- Time-lock with reasonable latency
- Assignment of privileged roles to multi-sig wallets to prevent single point of failure
- Introduction of a DAO / governance / voting module to increase transparency and user involvement.

Team Comment: The project is already modelled as a DAO, and LP will be voted on through governance.

WWI-1 – State variable visibility is not set

Severity: Low **Status**: Acknowledged

Description: The global state variable "*_balances*" visibility is unspecified. Labelling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

Risk: Minor informational issue, RE: CWE-710: Improper Adherence to Coding Standards.

Recommendation: It is best practice to set the visibility of all state variables explicitly. The default visibility for " *_balances* " is internal.

SWC Registry: SWC-108

WWI-2 – Unhandled return value

Severity: Low **Status**: Acknowledged

Description: The return value of the function call "*addLiquidityETH*" is not checked or handled.

Risk: Execution will resume even if the "*addLiquidityETH*" throws an exception. If the call fails accidentally or an attacker forces the call to fail, this may cause unexpected behaviour in the subsequent program logic.

Recommendation: We recommend using variable to receive the return value of the "*addLiquidityETH*" function call and handle both success and failure scenarios.

SWC Registry: SWC-104

WWI-3 – Missing zero-address validation

Severity: Low **Status**: Acknowledged

Description: The "*setmarketingAddress*", "*settreasuryAddress*" functions do not include any validation to ensure that the address being set isn't the zero or dead address.

Risk: While the risk is minimal, it is still a possibility for the marketing address to be set to the zero or dead addresses, causing loss of tokens and marketing funds from the project.

Recommendation: We recommend adding checks to ensure that the address provided as input to functions are valid addresses for its intended use.

WWI-4 – Environment & function/variable naming mismatch

Severity: Low **Status**: Acknowledged

Description: The WorldWarInu contract uses Pancakeswap for swapping and liquidity adds using BNB, however, functions and variable are named with Uniswap and Ethereum.

Risk: Mismatched function and variables names from the environment in which a smart contract operates can cause confusion.

Recommendation: We recommend changing Uniswap and ETH to Pancakeswap and BNB.

WWI-5 – Functions that could be declared external

Severity: Low **Status**: Acknowledged

Description: Several functions are declared as public visibility, however, since they are never called by the contract they should be declared external.

Risk: This is a gas optimization issue.

Recommendation: We recommend that functions that are never called by the contract to be declared as external to save gas.

Code Documentation

The code has a moderate amount of comments. This could be improved in order to help others understand the contract.

Adherence to Specifications

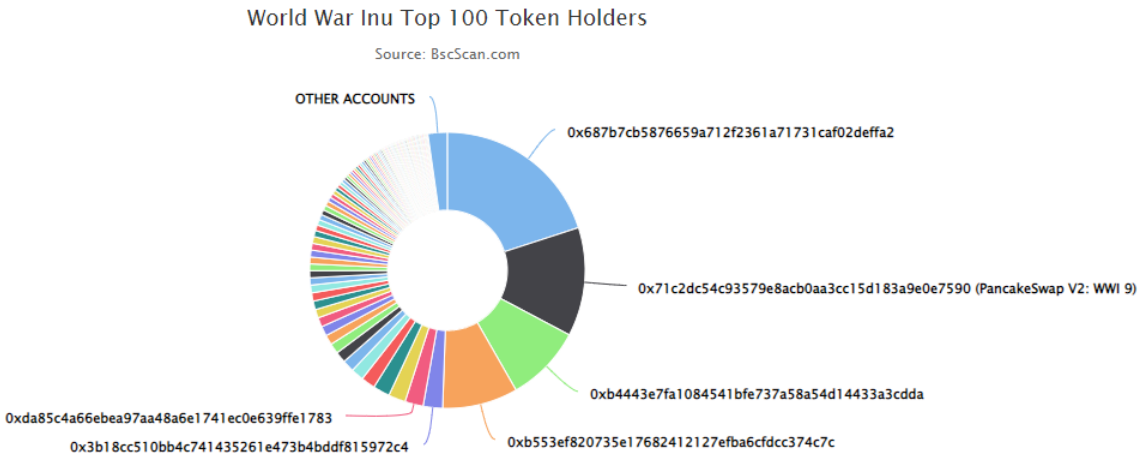
The smart contract adheres to the smart contract functionality described by the World War Inu team and is in line with its intended usage.

Adherence to Best Practices

The smart contract adheres to the majority of best practices associated with a standard BEP-20 token.

On-Chain Analysis

Token/Holder Distribution



- DAO controlled wallets are:
  - 0x687b7cb5876659a712f2361a71731caf02deffa2 - Treasury
  - 0xb553ef820735e17682412127efba6cfdcc374c7c - Marketing
  - 0xb4443e7fa1084541bfe737a58a54d14433a3cdda - Owner

- DAO controlled wallets contain clost to 40% of the token supply, which is acceptable as long as their distribution is decided upon by the DAO

Priviledged Transactions

Transactions from the Owner wallet seem to be distributions and airdrops to community members and also to provision liquidity.

Liquidity

- A total of 26.75 BNB is added to liquidity on PancakeSwap, all of which is controlled by the Owner wallet.

- Initial LP has been locked on DxSale: https://bscscan.com/tx/0xe664bd2ba192cc251b067e9283e0d31f43188064a312f2ef08358f5fafcc3c5e

- Addition LP accumulated via trading volume currently sits in the Owner wallet, left for DAO vote to decide on its fate.

- Liquidity currently sits at 13% of overall market cap, which is within an acceptable range and considered a healthy amount of liquidity for a project of this size.