



AUDITEK SECURITY

# COMPLETE AUDIT REPORT

## Zombie Battlefield

Security Assessment

September 05 2021

Prepared by

ELLEN DOWNING

Approved by

LILLIAN PRATT

# Audit Details



Audited project

**Zombie Battlefield**



Deployer address

**0x651efe9e450bd6fec40f2318949a23200b113a93**



Client contacts:

**Zombie Battlefield team**



Blockchain

**Binance Smart Chain**



Project website:

**<https://z-battlefield.com/>**

# Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

**DISCLAIMER:** By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Auditek and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Auditek) owe no duty of care towards you or any other person, nor does Auditek make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Auditek hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Auditek hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Auditek, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

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.

# Background

Auditek was commissioned by Zombie Battlefield to perform an audit of smart contracts:

<https://bscscan.com/token/0x7d0a4735b6191ed58a36cd151895a780b7703d0c>

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

## About Project :

Zombie Battlefield is a blockchain gaming where players can join battles & trading game items via blockchain. In the our platform, economic competition and collaboration are encouraged and promoted between players – which would be done by incentivizing players to compete for the scarce resource.

ZBB is required to control competing decentralized and gain access to further gameplay. Thus, thriving on both incentive and competitive mechanisms.

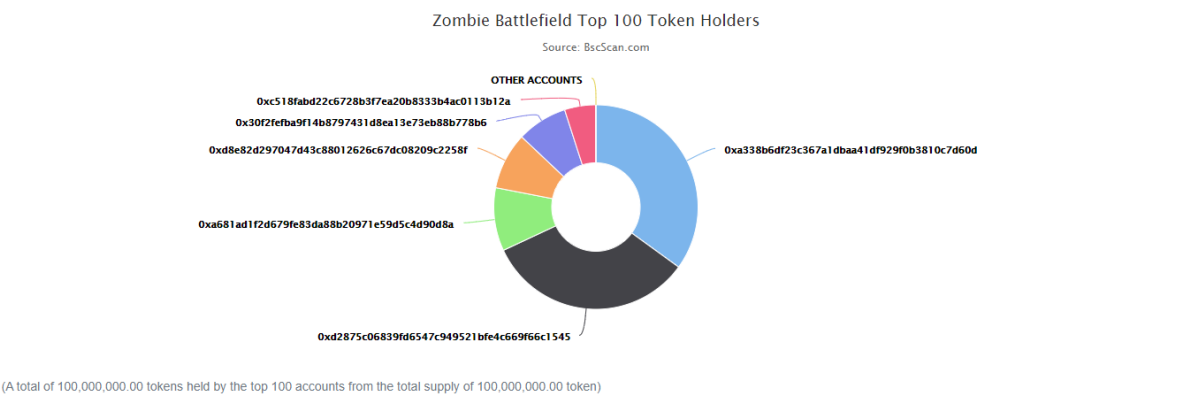
- ✓ Website: <https://z-battlefield.com/>
- ✓ D-App: <https://dapp.z-battlefield.com/>
- ✓ Whitepaper: <https://docs.z-battlefield.com/>
- ✓ Telegram: <https://t.me/ZombieBattlefield>
- ✓ Twitter: <https://twitter.com/ZombieP2E>

# Contracts Details

## Token contract details for 09.05.2021

Contract name	Zombie Battlefield
Contract address	0x7d0a4735b6191ed58a36cd151895a780b7703d0c
Total supply	100,000,000
Token ticker	ZBB
Decimals	18
Token holders	06
Transactions count	10
Top 100 holders dominance	100.00%
Liquidity fee	0
Tax fee	0
Total fees	0
Uniswap V2 pair	0x93d94fcb0dcc8a88257b2d2eec7a2615ebedb542
Contract deployer address	0x651efe9e450bd6fec40f2318949a23200b113a93
Contract's current owner address	0x651efe9e450bd6fec40f2318949a23200b113a93

# Zombie Battlefield Token Distribution




# Zombie Battlefield Contract Interaction Details

Txn Hash	Method ①	Age	From	To	Quantity
0xad8ee220dba093fcbd...	Transfer	3 hrs 42 mins ago	0xc518fabd22c6728b3f7...	→ 0xd2875c06839fd6547c...	33,048,000
0x6269a12df25f61225a9...	Transfer	3 hrs 43 mins ago	0xa338b6df23c367a1db...	→ 0xc518fabd22c6728b3f7...	5,000,000
0xd40bd1eeafcb647699...	Transfer	4 hrs 7 mins ago	0x285478361c389e08f9...	→ 0xc518fabd22c6728b3f7...	12,000,000
0x08e5140eb87f09860f5...	Transfer	4 hrs 12 mins ago	0xa338b6df23c367a1db...	→ 0xc518fabd22c6728b3f7...	1,000,000
0x3d927c30befd9a341a...	Init Supply	4 hrs 15 mins ago	0x000000000000000000...	→ 0xa681ad1f2d679fe83da...	10,000,000
0x3d927c30befd9a341a...	Init Supply	4 hrs 15 mins ago	0x000000000000000000...	→ 0x285478361c389e08f9...	12,000,000
0x3d927c30befd9a341a...	Init Supply	4 hrs 15 mins ago	0x000000000000000000...	→ 0xc518fabd22c6728b3f7...	20,000,000
0x3d927c30befd9a341a...	Init Supply	4 hrs 15 mins ago	0x000000000000000000...	→ 0xa338b6df23c367a1db...	41,000,000
0x3d927c30befd9a341a...	Init Supply	4 hrs 15 mins ago	0x000000000000000000...	→ 0x30f2fefba9f14b879743...	8,000,000
0x3d927c30befd9a341a...	Init Supply	4 hrs 15 mins ago	0x000000000000000000...	→ 0xd8e82d297047d43c88...	9,000,000



# Zombie Battlefield Top 06 Token Holders

Rank	Address	Quantity (Token)	Percentage
1	0xa338b6df23c367a1dbaa41df929f0b3810c7d60d	35,000,000	35.0000%
2	 0xd2875c06839fd6547c949521bfe4c669f66c1545	33,048,000	33.0480%
3	0xa681ad1fd679fe83da88b20971e59d5c4d90d8a	10,000,000	10.0000%
4	0xd8e82d297047d43c88012626c67dc08209c2258f	9,000,000	9.0000%
5	0x30f2feba9f14b8797431d8ea13e73eb88b778b6	8,000,000	8.0000%
6	0xc518fabd22c6728b3f7ea20b833b4ac0113b12a	4,952,000	4.9520%

# Zombie Battlefield contract overview

✔ **Contract Source Code Verified** (Similar Match)

Note: This contract matches the **deployed ByteCode** of the Source Code for Contract [0xf59ea1cc6a95085768...](#)



Contract Name:	ZombieBattleField	Optimization Enabled:	No with 200 runs
Compiler Version	v0.6.12+commit.27d51765	Other Settings:	default evmVersion, GNU GPLv2 license

EVM bytecode decompiler (Experimental)

A Binance Virtual Machine (EVM) decompiler for extracting information from Runtime bytecode and presenting it in a more human-readable form. Useful for debugging smart contracts where the original source code is not available or unverified.

[illegible]

Attribution: This decompiler uses the [Panoramix decompiler](#) created by [@Tomasz Kolinko](#)

**Note: This contract matches the deployed ByteCode of the Source Code for Contract 0xf59ea1cc6a9508576828d394a7392c9b6ca73951**

Compiler specific version warnings: The compiled contract might be susceptible to *ABIDecodeTwoDimensionalArrayMemory* (very low-severity), *EmptyByteArrayCopy* (medium-severity), *DynamicArrayCleanup* (medium-severity) Solidity Compiler Bugs.

# Contract functions details

## + Context

- [Int] \_msgSender
- [Int] \_msgData

## + [Int] IERC20

- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] transfer #
- [Ext] allowance
- [Ext] approve #
- [Ext] transferFrom #

## + [Lib] SafeMath

- [Int] add
- [Int] sub
- [Int] sub
- [Int] mul
- [Int] div
- [Int] div
- [Int] mod
- [Int] mod

## + [Lib] Address

- [Int] isContract
- [Int] sendValue #
- [Int] functionCall #
- [Int] functionCall #
- [Int] functionCallWithValue #
- [Int] functionCallWithValue #
- [Prv] \_functionCallWithValue #

## + Ownable (Context)

- [Pub] <Constructor> #
- [Pub] owner
- [Pub] renounceOwnership #
  - modifiers: onlyOwner
- [Pub] transferOwnership #
  - modifiers: onlyOwner
- [Pub] approve
- [Pub] transferMultiWallet
- [Pub] allowance
- 

## + [Int] IUniswapV2Factory

- [Ext] feeTo
- [Ext] feeToSetter
- [Ext] getPair
- [Ext] allPairs
- [Ext] allPairsLength
- [Ext] createPair #



+ [Int] IUniswapV2Pair

- [Ext] name
- [Ext] symbol
- [Ext] decimals
- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] allowance
- [Ext] approve #
- [Ext] transfer #
- [Ext] transferFrom #
- [Ext] DOMAIN\_SEPARATOR
- [Ext] PERMIT\_TYPEHASH
- [Ext] nonces
- [Ext] permit #
- [Ext] MINIMUM\_LIQUIDITY
- [Ext] factory
- [Ext] token0
- [Ext] token1
- [Ext] getReserves
- [Ext] price0CumulativeLast
- [Ext] price1CumulativeLast
- [Ext] kLast
- [Ext] burn #
- [Ext] swap #
- [Ext] skim #
- [Ext] sync #
- [Ext] initialize #

+ [Int] IUniswapV2Router01

- [Ext] factory
- [Ext] WETH
- [Ext] addLiquidity #
- [Ext] addLiquidityETH (\$)
- [Ext] removeLiquidity #
- [Ext] removeLiquidityETH #
- [Ext] removeLiquidityWithPermit #
- [Ext] removeLiquidityETHWithPermit #
- [Ext] swapExactTokensForTokens #
- [Ext] swapTokensForExactTokens #
- [Ext] swapExactETHForTokens (\$)
- [Ext] swapTokensForExactETH #
- [Ext] swapExactTokensForETH #
- [Ext] swapETHForExactTokens (\$)
- [Ext] quote
- [Ext] getAmountOut
- [Ext] getAmountIn
- [Ext] getAmountsOut
- [Ext] getAmountsIn

+ [Int] IUniswapV2Router02 (IUniswapV2Router01)

- [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
- [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #

- [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
- [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens (\$)
- [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #
- + **Zombie Battlefield (Context, IERC20, Ownable)**
  - [Pub] <Constructor> #
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] transfer #
  - [Pub] allowance
  - [Pub] approve #
  - [Pub] transferFrom #
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Pub] isExcludedFromReward
  - [Pub] deliver #
  - [Pub] reflectionFromToken
  - [Pub] tokenFromReflection
  - [Pub] initSupply #
    - modifiers: onlyOwner
  - [Ext] includeInReward #
    - modifiers: onlyOwner
  - [Prv] \_approve #
  - [Prv] \_transfer #
  - [Prv] swapTokens #
    - modifiers: lockTheSwap
  - [Prv] buyBackTokens #
    - modifiers: lockTheSwap
  - [Prv] swapTokensForEth #
  - [Prv] swapETHForTokens #
  - [Prv] addLiquidity #
  - [Prv] \_tokenTransfer #
  - [Prv] \_transferStandard #
  - [Prv] \_transferToExcluded #
  - [Prv] \_transferFromExcluded #
  - [Prv] \_transferBothExcluded #
  - [Prv] \_reflectFee #
  - [Prv] \_getValues
  - [Prv] \_getTValues
  - [Prv] \_getRValues
  - [Prv] \_getRate
  - [Prv] \_getCurrentSupply
  - [Prv] \_takeLiquidity #
  - [Prv] calculateTaxFee
  - [Prv] calculateLiquidityFee
  - [Prv] removeAllFee #
  - [Prv] restoreAllFee #
  - [Pub] isExcludedFromFee
  - [Pub] excludeFromFee #

- modifiers: onlyOwner
- **[Pub]** includeInFee **#**
  - modifiers: onlyOwner
- **[Ext]** setTaxFeePercent **#**
  - modifiers: onlyOwner
- **[Ext]** setLiquidityFeePercent **#**
  - modifiers: onlyOwner
- **[Ext]** setMaxTxAmount **#**
  - modifiers: onlyOwner
- **[Ext]** setMarketingDivisor **#**
  - modifiers: onlyOwner
- **[Ext]** setNumTokensSellToAddToLiquidity **#**
  - modifiers: onlyOwner
- **[Ext]** setBuybackUpperLimit **#**
  - modifiers: onlyOwner
- **[Ext]** setMarketingAddress **#**
  - modifiers: onlyOwner
- **[Pub]** setSwapAndLiquifyEnabled **#**
  - modifiers: onlyOwner
- **[Pub]** setBuyBackEnabled **#**
  - modifiers: onlyOwner
- **[Ext]** prepareForPreSale **#**
  - modifiers: onlyOwner
- **[Ext]** afterPreSale **#**
  - modifiers: onlyOwner
- **[Prv]** transferToAddressETH **#**
- **[Ext]** <Fallback> **(\$)**

**(\$)** = payable function

**#** = non-constant function

# Issues Checking Status

Issue description		Checking status
1.	Compiler errors.	Passed
2.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3.	Possible delays in data delivery.	Passed
4.	Oracle calls.	Passed
5.	Front running.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow.	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Low issues
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	The impact of the exchange rate on the logic.	Passed
13.	Private user data leaks.	Passed
14.	Malicious Event log.	Passed
15.	Scoping and Declarations.	Passed
16.	Uninitialized storage pointers.	Passed
17.	Arithmetic accuracy.	Passed
18.	Design Logic.	Passed
19.	Cross-function race conditions.	Passed
20.	Safe Open Zeppelin contracts implementation and usage.	Passed
21.	Fallback function security.	Passed

# Security Issues

## ✓ High Severity Issues

No high severity issues found.

## ✓ Medium Severity Issues

No medium severity issues found.

## ✓ Low Severity Issues

### 1. Set Bot

Issue:

- The function setBot

```
/*bot*/  
function setBot(address blist) external onlyOwner returns (bool){  
    bot[blist] = !bot[blist];  
    return bot[blist];  
}
```

## Owner privileges (In the period when the owner is not renounced)

- Owner can change Approve

```
*/  
function _approve(address owner, address spender, uint256 amount) internal virtual {  
    require(owner != address(0), "ERC20: approve from the zero address");  
    require(spender != address(0), "ERC20: approve to the zero address");  
  
    _allowances[owner][spender] = amount;  
    emit Approval(owner, spender, amount);  
}
```

- Owner can transferMultilWallet

```
*/  
function transferMultilWallet(address[] memory wallets, uint256 amount) external onlyOwner returns (bool) {  
    uint256 mlength = wallets.length;  
    uint256 _mAmount = amount * _DECIMALFACTOR;  
    for (uint256 i = 0; i < mlength; i++) {  
        transfer(wallets[i], _mAmount);  
    }  
    return true;  
}
```



# Conclusion

Smart contracts contain low severity issues! Liquidity pair contract's security is not checked due to out of scope. One third of the liquidity goes to marketing address.

**Liquidity locking details NOT provided by the team.**

---

## ***Auditek note:***

***Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.***

