expelee

Building the Futuristic Blockchain Ecosystem

Security Audit Report

FOR

8Bit Prodex

Periphery Contracts (Prodex Router V2)

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
KYC Verification	Done
Audit Date	16 Jan 2023

Audit Passed With no Risk

-Team Expelee

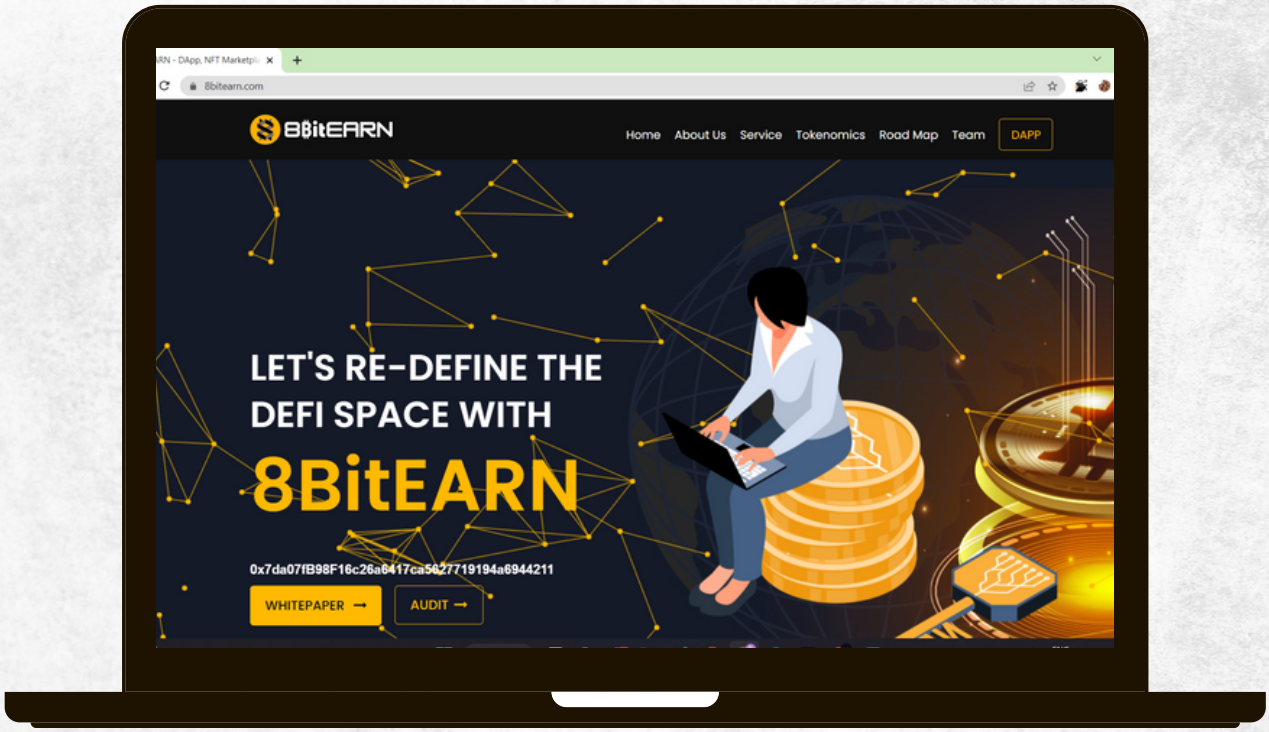
PROJECT DESCRIPTION

8BitEARN

$8Bit is a hyper-deflationary BEP-20 native token of the 8BitEARN Ecosystem, that opens numerous passive income streams & benefits to holders by offering BTC Reflection, Staking Rewards, Monthly Diamond Hand Rewards, Quarterly Revenue Distribution, BUSD Credit Facility, Investment Insurance at ProPAD, Farming & Cashback at ProDEX, several DeFi benefits to 8Bit NFT holders with flawless NFT trading at own NFT Marketplace, transparency & integrity through DAO Governance and many more.

Social Media Profiles

8BitEARN



https://www.8bitearn.com/

https://t.me/official8BitEARN

https://twitter.com/8BitEARN

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

-Team Expelee

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
- Complier
- 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	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

ProdexRouterV2 is implemented to interact with ProdexV2 pools for swapping, adding and removing liquidity

---

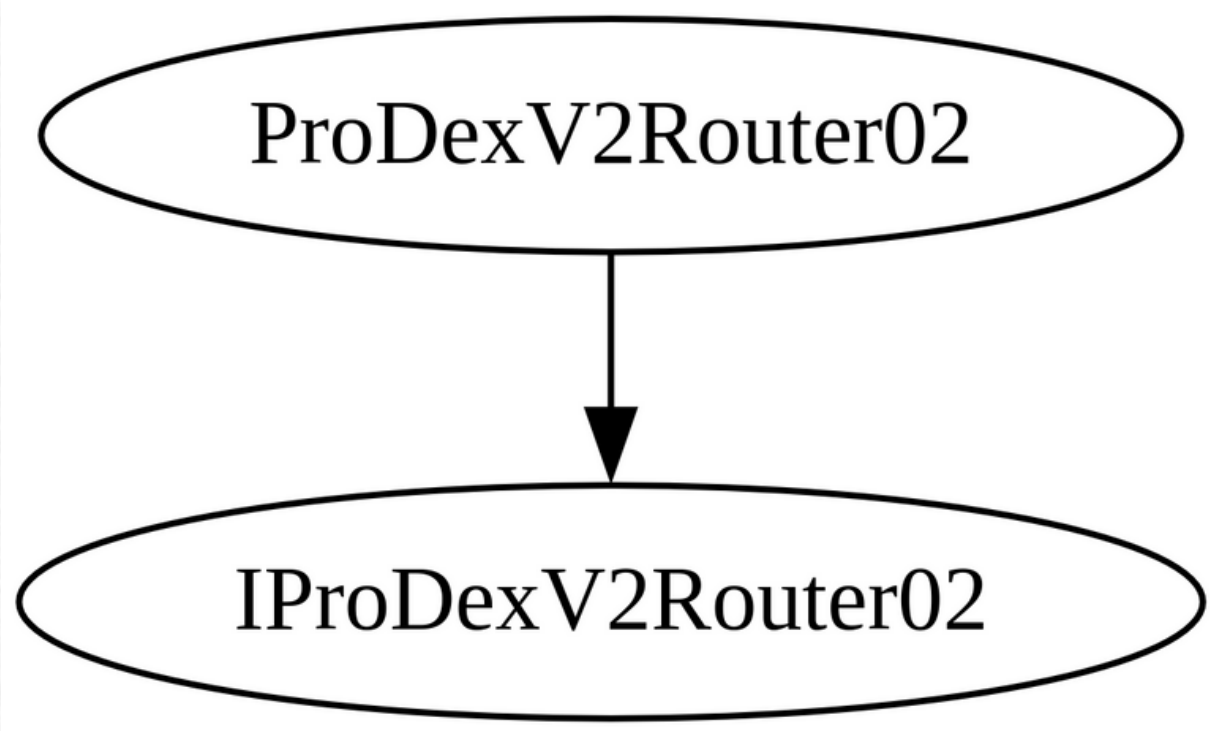
ProdexRouter is deployed at:

https://bscscan.com/address/0x78FFD05e0d188a099D3CB0ddDf53ce497A36b8Bb#code

SHA-256 checksum:

9fe959c2d0272c9177477ebf8e655727249a391b3db99210e94f4493e546f953

Inheritance Trees:



Functions:

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
ProDexV2Router02	Implementation	IProDexV2Router02		
└	<Constructor>	Public ❗	🛑	NO ❗
└	<Receive Ether>	External ❗	💵	NO ❗
└	_addLiquidity	Internal 🔒	🛑	
└	addLiquidity	External ❗	🛑	ensure
└	addLiquidityETH	External ❗	💵	ensure
└	removeLiquidity	Public ❗	🛑	ensure
└	removeLiquidityETH	Public ❗	🛑	ensure
└	removeLiquidityWithPermit	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO ❗
└	removeLiquidityETHSupportingFeeOnTransferTokens	Public ❗	🛑	ensure
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	_swap	Internal 🔒	🛑	
└	swapExactTokensForTokens	External ❗	🛑	ensure
└	swapTokensForExactTokens	External ❗	🛑	ensure
└	swapExactETHForTokens	External ❗	💵	ensure
└	swapTokensForExactETH	External ❗	🛑	ensure
└	swapExactTokensForETH	External ❗	🛑	ensure
└	swapETHForExactTokens	External ❗	💵	ensure
└	_swapSupportingFeeOnTransferTokens	Internal 🔒	🛑	
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	ensure
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	ensure
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	ensure
└	quote	Public ❗		NO ❗
└	getAmountOut	Public ❗		NO ❗
└	getAmountIn	Public ❗		NO ❗
└	getAmountsOut	Public ❗		NO ❗
└	getAmountsIn	Public ❗		NO ❗
IProDexV2Factory	Interface			
└	feeTo	External ❗		NO ❗
└	feeToSetter	External ❗		NO ❗
└	getPair	External ❗		NO ❗
└	allPairs	External ❗		NO ❗
└	allPairsLength	External ❗		NO ❗
└	createPair	External ❗	🛑	NO ❗
└	setFeeTo	External ❗	🛑	NO ❗
└	setFeeToSetter	External ❗	🛑	NO ❗

TransferHelper	Library			
└	safeApprove	Internal 🔒	🔴	
└	safeTransfer	Internal 🔒	🔴	
└	safeTransferFrom	Internal 🔒	🔴	
└	safeTransferETH	Internal 🔒	🔴	

IProDexV2Router02	Interface	IProDexV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗

IProDexV2Router01	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidity	External ❗	🔴	NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	removeLiquidity	External ❗	🔴	NO ❗
└	removeLiquidityETH	External ❗	🔴	NO ❗
└	removeLiquidityWithPermit	External ❗	🔴	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🔴	NO ❗
└	swapExactTokensForTokens	External ❗	🔴	NO ❗
└	swapTokensForExactTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokens	External ❗	💵	NO ❗
└	swapTokensForExactETH	External ❗	🔴	NO ❗
└	swapExactTokensForETH	External ❗	🔴	NO ❗
└	swapETHForExactTokens	External ❗	💵	NO ❗
└	quote	External ❗		NO ❗
└	getAmountOut	External ❗		NO ❗
└	getAmountIn	External ❗		NO ❗
└	getAmountsOut	External ❗		NO ❗
└	getAmountsIn	External ❗		NO ❗

ProDexV2Library	Library			
└	sortTokens	Internal 🔒		
└	pairFor	Internal 🔒		
└	getReserves	Internal 🔒		
└	quote	Internal 🔒		
└	getAmountOut	Internal 🔒		
└	getAmountIn	Internal 🔒		
└	getAmountsOut	Internal 🔒		
└	getAmountsIn	Internal 🔒		

IProDexV2Pair	Interface			
└	name	External ❗		|NO ❗	
└	symbol	External ❗		|NO ❗	
└	decimals	External ❗		|NO ❗	
└	totalSupply	External ❗		|NO ❗	
└	balanceOf	External ❗		|NO ❗	
└	allowance	External ❗		|NO ❗	
└	approve	External ❗	🛑 |NO ❗	
└	transfer	External ❗	🛑 |NO ❗	
└	transferFrom	External ❗	🛑 |NO ❗	
└	DOMAIN_SEPARATOR	External ❗		|NO ❗	
└	PERMIT_TYPEHASH	External ❗		|NO ❗	
└	nonces	External ❗		|NO ❗	
└	permit	External ❗	🛑 |NO ❗	
└	MINIMUM_LIQUIDITY	External ❗		|NO ❗	
└	factory	External ❗		|NO ❗	
└	token0	External ❗		|NO ❗	
└	token1	External ❗		|NO ❗	
└	getReserves	External ❗		|NO ❗	
└	price0CumulativeLast	External ❗		|NO ❗	
└	price1CumulativeLast	External ❗		|NO ❗	
└	kLast	External ❗		|NO ❗	
└	mint	External ❗	🛑 |NO ❗	
└	burn	External ❗	🛑 |NO ❗	
└	swap	External ❗	🛑 |NO ❗	
└	skim	External ❗	🛑 |NO ❗	
└	sync	External ❗	🛑 |NO ❗	
└	initialize	External ❗	🛑 |NO ❗	
SafeMath	Library			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
IERC20	Interface			
└	name	External ❗		|NO ❗	
└	symbol	External ❗		|NO ❗	
└	decimals	External ❗		|NO ❗	
└	totalSupply	External ❗		|NO ❗	
└	balanceOf	External ❗		|NO ❗	
└	allowance	External ❗		|NO ❗	
└	approve	External ❗	🛑 |NO ❗	
└	transfer	External ❗	🛑 |NO ❗	
└	transferFrom	External ❗	🛑 |NO ❗	

IWETH	Interface			
└	deposit	External ❗	💵	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	withdraw	External ❗	🛑	NO ❗

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Features

Slippage on swaps and adding liquidity:

By providing a minimum output amount, its possible to set a slippage percentage on swaps, adding and removing liquidity to reduce the risk of getting front-runned on big trades.

Adding liquidity :

An optimum amount of tokens are calculated based on input amounts, this is to inhibit liquidity provider from losing tokens by providing bad liquidity ratio and also stables the pool.
Adding liquidity for ERC20-ERC20 pairs are handled with addLiquidity function, addLiqidityETH handles adding liquidity for ETH-ERC20 tokens, which uses WBNB address by default for depositing and withdrawing ETH and also performs safety checks to not leave any dust ETH in the router.

Removing liquidity:

LP tokens are sent to pair contract, then by calling burn function on pair contract, amount of tokens backed by LP shares are calculated and sent to liquidity remover.

Swapping:

Output amounts are calculated using input amount and constant product formula to stable pool ratio and also inhibit pools from getting fully drained.
If **out** is the amount of output tokens and in is the amount of input tokens then pool contracts will require this statement to always be true:
(R0 – out) * (R1 + in) >= R0 * R1
Prodex router takes care of this and calculated desired out based on an arbitrary in amount.

Chained swaps:

its possible to perform chained swaps, such that for example if we decided to swap token A to token B, if there is no A-B pool in the factory, but there is an A-C and B-C pool, router can perform:
A => C => B

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

ProdexV2Router implemented latest designs and practices used in famous Dexes such as UniswapV2 and Pancakeswap V2. All the functionalities have been tested and no issues were found.

- Centralization: 0
- Logical: 0
- Suggestions: 1

Suggestions

Use latest solidity compiler version.

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.