expelee

A Secure Place For Web3

SMART CONTRACT AUDIT OF

DOGE BET COIN

Contract Address

0x557715C2e809D99fac7F632835f65442a17C35a4

Audit Summary

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: NOT DONE

Audit Date: 26/08/2022

Audit Team: **EXPELEE**

Be aware that smart contracts deployed on the blockchain aren’t resistant to internal exploit, external vulnerability, or hack. For a detailed understanding of risk severity, source code vulnerability, functional hack, and audit disclaimer, kindly refer to the audit.

DISCLAMER

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.

Contract Review

Contract Name	DOGE BET COIN
Compiler Version	v0.7.6+commit.7338295f
Optimization	No with 200 runs
License	default license
Explorer	https://explorer.dogechain.dog/address/0x557715C2e809D99fac7F632835f65442a17C35a4/contracts
Symbol	DFC
Decimals	18
Total Supply	10,000,000,000 DBC
Domain	https://dogebetcoin.com/

Project Review

Token Name: DOGE BETCOIN

Web Site: https://dogebetcoin.com/

Twitter: https://twitter.com/dogebetcoin

Telegram: https://t.me/dogebetcoins

Contract Address:

0x557715C2e809D99fac7F632835f65442a17C35a4

Platform: DOGE CHAIN

Token Type: ERC 20

Language: SOLIDITY

Audit Methodology

The scope of this report is to audit the smart contract source code. We have scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Category

Smart Contract Vulnerabilities	- Unhandled Exceptions - Transaction Order Dependency - Integer Overflow - Unrestricted Action - Incorrect Inheritance Order - Typographical Errors - Requirement Violation
Source Code Review	- Gas Limit and Loops - Deployment Consistency - Repository Consistency - Data Consistency - Token Supply Manipulation
Functional Assessment	- Operations Trail & Event Generation - Assets Manipulation - Liquidity Access

Vulnerability Checklist

Nº	Description.	Result
1	Compiler warnings.	Passed
2	Race conditions and Re-entrancy. Cross-function raceconditions.	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.	Passed
10	Methods execution permissions.	Passed
11	Economy model.	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 Zeppelin module.	Passed
21	Fallback function security.	Passed

Manual Audit

🟢**Low-Risk RISK LEVEL(2-4)**

🟢2 low-risk code issues found

🟡**Medium-Risk RISK LEVEL(4-7)**

🟡1 medium-risk code issues found

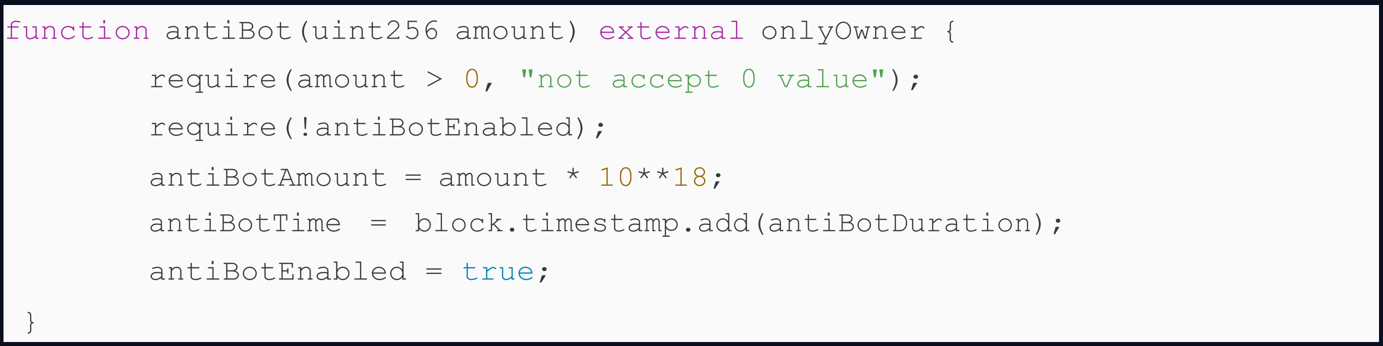
🔴**High-Risk RISK LEVEL(7-10)**

🔴0 high-risk code issues found

MANUAL AUDIT

● Medium:

enabling anti-bot with an undesired trade size limit



```
function antiBot(uint256 amount) external onlyOwner {
        require(amount > 0, "not accept 0 value");
        require(!antiBotEnabled);
        antiBotAmount = amount * 10**18;
        antiBotTime  =  block.timestamp.add(antiBotDuration);
        antiBotEnabled = true;
 }
```

anti-bot is not enabled yet at time of writing this audit, but owner is able to use **antiBot** function to enable anti-bot with a specified limit for trade size (its a 1 time use function & after enabling anti-bot,owner is not able to use this function again), this means that if a **bot** try to sell or buy in anti-bot duration and his buying or selling amount is more than anti-bot limit, the transaction will revert. **there is 2 concerns:**

1- setting trading limit to a very little number (0) causes bots to not be able to trade, this is considered acentralization issue because owner is able to set an arbitrary address as **bot** and disable all of its trades.

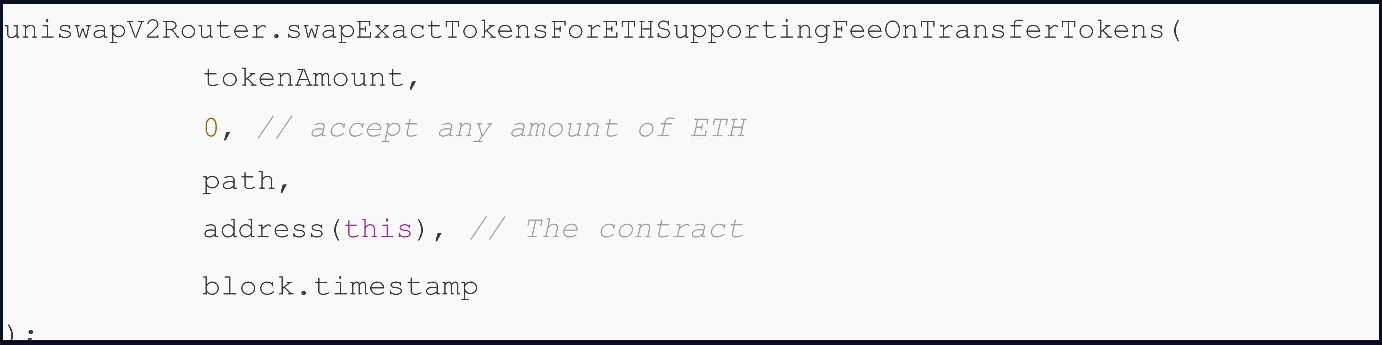
2- owner may accidently set an undesired trade size limit and since this function enables anti-bot and isonly 1 time use, there is no way to change trading limit again.

Recommendation:

remove this function and set a logic inside _transfer functoin to automatically start & end(forever) anti-bot based on block number or timestamp.

● Low :

Line-871 => potential sandwitch attack



```
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            tokenAmount,
            0, // accept any amount of ETH
            path,
            address(this), // The contract
            block.timestamp
);
```

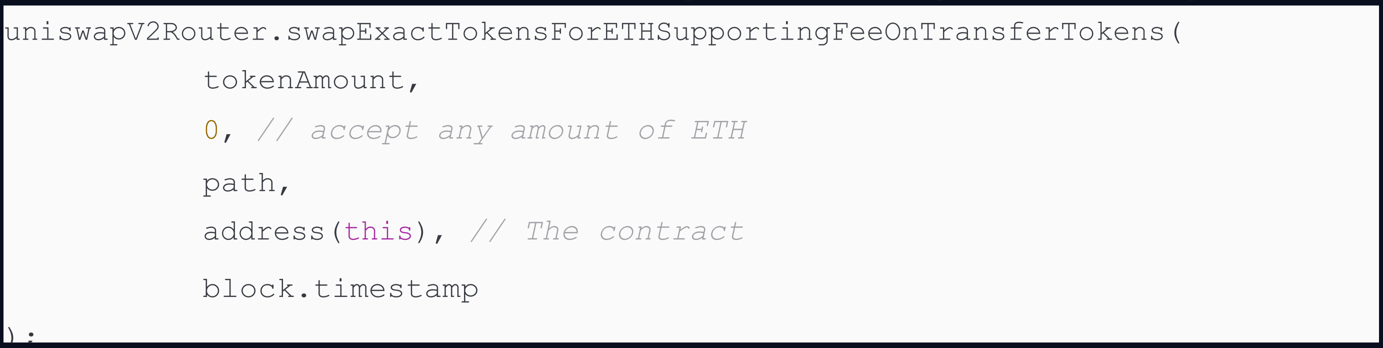
A sandwich attack might happen when an attacker observesa transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulatethe exchange rate by frontrunning (before the transaction being attacked) a transaction to purchase one ofthe assets and make profits by backrunning (after the transaction being attacked) a transaction to sell theasset.

Recommendation:

give a reasonable output amount based on price

● Low :

Line-871 => Lack of returns value handling



```
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            tokenAmount,
            0, // accept any amount of ETH
            path,
            address(this), // The contract
            block.timestamp
);
```

Return value (true or false) is not being handled her

Recommendation:

We recommend using variables to receive the return value of
the functions mentioned above and
handleboth success and failure cases if needed by the
business logic

Important Points To Consider

Owner Change Balance	Not detected
Blacklist	Not detected
Modify Fees	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

About Expelee

Expelee is a community driven organisation dedicated to fostering an anti-rug movement. We're here to keep investment safe from fraudsters. We've encountered several rug pulls and know how it feels to be duped, which is why we don't want anybody else to go through the same experience. We are here to raise awareness through our services so that the future of cryptocurrency can be rug-free.

The auditing process focuses to the following considerations with collaboration of an expert team:

- Functionality test of the Smart Contract to determine if proper logic has been followed throughout the whole process.
- Manually detailed examination of the code line by line by experts.
- Live test by multiple clients using Test net.
- Analysing failure preparations to check how the Smart
- Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analysing the security of the on-chain data.

Social Media

/Expelee

/ExpeleeOfficial

/expelee-co