InterFi
Network

@interfinetwork

SMART CONTRACT SECURITY AUDIT

Daemon Doge

DAEMON
DOGE

AUDITED ON AUGUST 31, 2021

USING INTERFI AUDITING ARCHITECTURE

Summary

Audit:

Auditing Firm	InterFi Network
Architecture	InterFi Auditing Architecture
Smart Contract Audit Approved By	Chris | Blockchain Specialist at InterFi
Project Overview Approved BY	Albert | Project Specialist at InterFi
Platform	Solidity / BSC
Audit Check (Mandatory)	Vulnerability Check, Source Code Review, Functional Test
Project Check (Optional)	Website Review, Socials Review, Token Review (Not Applicable)
Consultation Request Date	August 28, 2021
Report Date	August 31, 2021

Risk profile:

InterFi team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit, **Daemon Doge's smart contract source code has Low Risk Severity.**

For the detailed understanding of risk severity, source code vulnerability, and functional test, kindly refer to the audit. At the time of the audit, the contract is not deployed on any blockchain. Note, the owner/developer can change/modify contract before blockchain deployment. Please proceed with caution.

Table of contents

Project overview

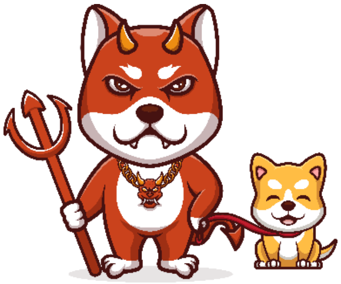
InterFi was consulted by Daemon Doge on August 28, 2021 to conduct a smart contract security audit of their solidity source code.

Public information

Daemon Doge is more than just a meme token, it is the first cross-chain community token to ever hit our crypto world. Their mission is simple, to create a sustainable wealth-generating token with no ulterior motives. Daemon Doge's vision is to build an ecosystem for its community that goes beyond the typical rewards system. Our goals will range from launching a Defi exchange, yield farming, and most importantly to create a token that provides value with continuous growth!

Information	Daemon Doge
Blockchain	No deployment info at the time of audit
Language	Solidity
Contract	https://github.com/Moe-B/DemonDoge/blob/main/contracts/DeamonDoge.sol
Website	https://daemondoge.com/
Twitter	https://twitter.com/DaemonDoge
Telegram	https://daemondoge.com/
Reddit	http://www.reddit.com/u/daemondoge
Medium	https://medium.com/@DaemonDoge
Instagram	https://www.instagram.com/daemon.doge/
GitHub	https://github.com/Moe-B/DemonDoge

Public logo



Audit scope and methodology

The scope of this report is to audit the smart contract source code of Daemon Doge. The source code can be viewed in its entirety on

https://github.com/Moe-B/DemonDoge/blob/main/contracts/DeamonDoge.sol

InterFi has 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	❖ Re-entrancy (RE)
	❖ Unhandled Exceptions (UE)
	❖ Transaction Order Dependency (TO)
	❖ Integer Overflow (IO)
	❖ Unrestricted Action (UA)
Source Code Review	❖ Ownership Takeover
	❖ Gas Limit and Loops
	❖ Deployment Consistency
	❖ Repository Consistency
	❖ Data Consistency
	❖ Code Typo Error
	❖ Token Supply Manipulation
Functional Assessment	❖ Access Control and Authorization
	❖ Operations Trail and Event Generation
	❖ Assets Manipulation
	❖ Liquidity Access

InterFi methodology

The aim of this report is to analyze the smart contract and identify the vulnerabilities and the hacks in the smart contract. Mentioned are the steps used by InterFi to assess the smart contract:

1. Code review that includes the following
   - Review of the specifications, sources, and instructions provided to InterFi to make sure we understand the size, scope, and functionality of the smart contract.
   - Manual review of code, which is the process of reading source code line-byline to identify potential vulnerabilities.
2. Testing and automated analysis that includes the following
   - Test coverage analysis, which is the process of determining whether the test cases are covering the code and how much code is exercised when we run those test cases.
   - Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts

Automated 3P frameworks used to assess the vulnerabilities

- Slither
- MythX
- Uniswap V2
- Open Zeppelin
- Solidity Code Complier

General risk factors

Smart contracts are generally designed to manipulate and hold funds denominated in Ether. This makes them very tempting attack targets, as a successful attack may allow the attacker to directly steal funds from the contract. Below are the typical risk levels of a smart contract:

Vulnerable: A contract is vulnerable if it has been flagged by a static analysis tool as such. As we will see later, this means that some contracts may be vulnerable because of a false-positive.

Exploitable: A contract is exploitable if it is vulnerable and the vulnerability could be exploited by an external attacker. For example, if the "vulnerability" flagged by a tool is in a function which requires to own the contract, it would be vulnerable but not exploitable.

Exploited: A contract is exploited if it received a transaction on Ethereum's main network which triggered one of its vulnerabilities. Therefore, a contract can be vulnerable or even exploitable without having been exploited.

Risk severity	Meaning
! Critical	This level vulnerabilities could be exploited easily, and can lead to asset loss, data loss, asset manipulation, or data manipulation. They should be fixed right away.
! High	This level vulnerabilities are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to critical risk severity
! Medium	This level vulnerabilities are should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution.
! Low	This level vulnerabilities can be ignored. They are code style violations, and informational statements in the code. They may not affect the smart contract execution

Audit overview

Knickknacks in the smart contract

Query	Result
maxSellTransactionAmount	100 * (10**6) * (10**18)
swapTokensAtAmount	20 * (10**6) * (10**18)
_maxWalletToken	15 * (10**9) * (10**18)
BNBRewardsFee	10
liquidityFee	3
marketingFee	2
buyingTotalFees	15
sellingBNBRewardsFee	20
sellingLiquidityFee	3
sellingMarketingFee	2
sellingTotalFees	25
gasForProcessing	300000
uniswapV2Router	0x9Ac64Cc6e4415144C455BD8E4837Fea55603e5c3
marketingWallet	0x483a332876694bFA0da7F683c09C1bF8f16B833c
swapAndLiquifyEnabled	True
symbol	DDOGE
Name	Daemon Doge
totalSupply	Not Available

Verifying token functions

Function	Description	Tested	Verdict
TotalSupply	provides information about the total token supply	**Yes**	**Passed**
BalanceOf	provides account balance of the owner's account	**Yes**	**Passed**
Transfer	executes transfers of a specified number of tokens to a specified address	**Yes**	**Passed**
TransferFrom	executes transfers of a specified number of tokens from a specified address	**Yes**	**Passed**
Approve	allow a spender to withdraw a set number of tokens from a specified account	**Yes**	**Passed**
Allowance	returns a set number of tokens from a spender to the owner	**Yes**	**Passed**

Not Verified

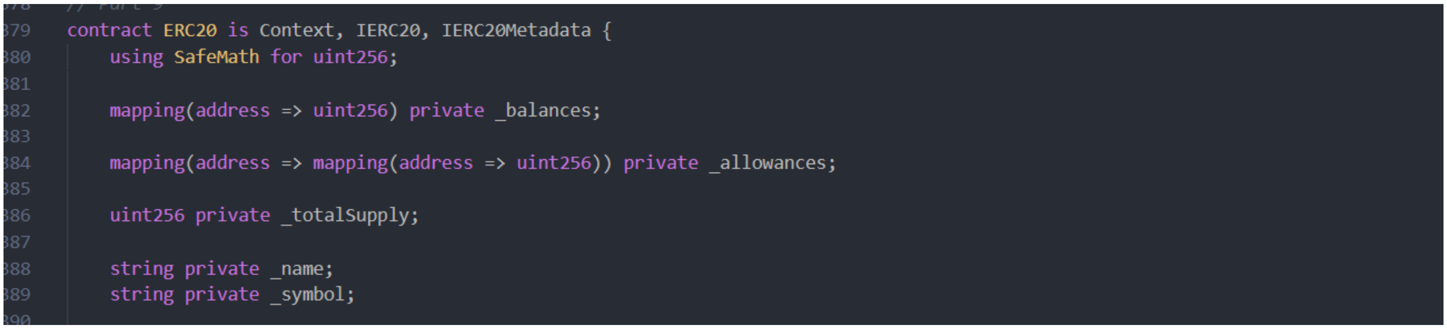
- Owner can not mint new tokens

Verified

- Owner can not pause the contract
- Owner can not burn/lock user assets

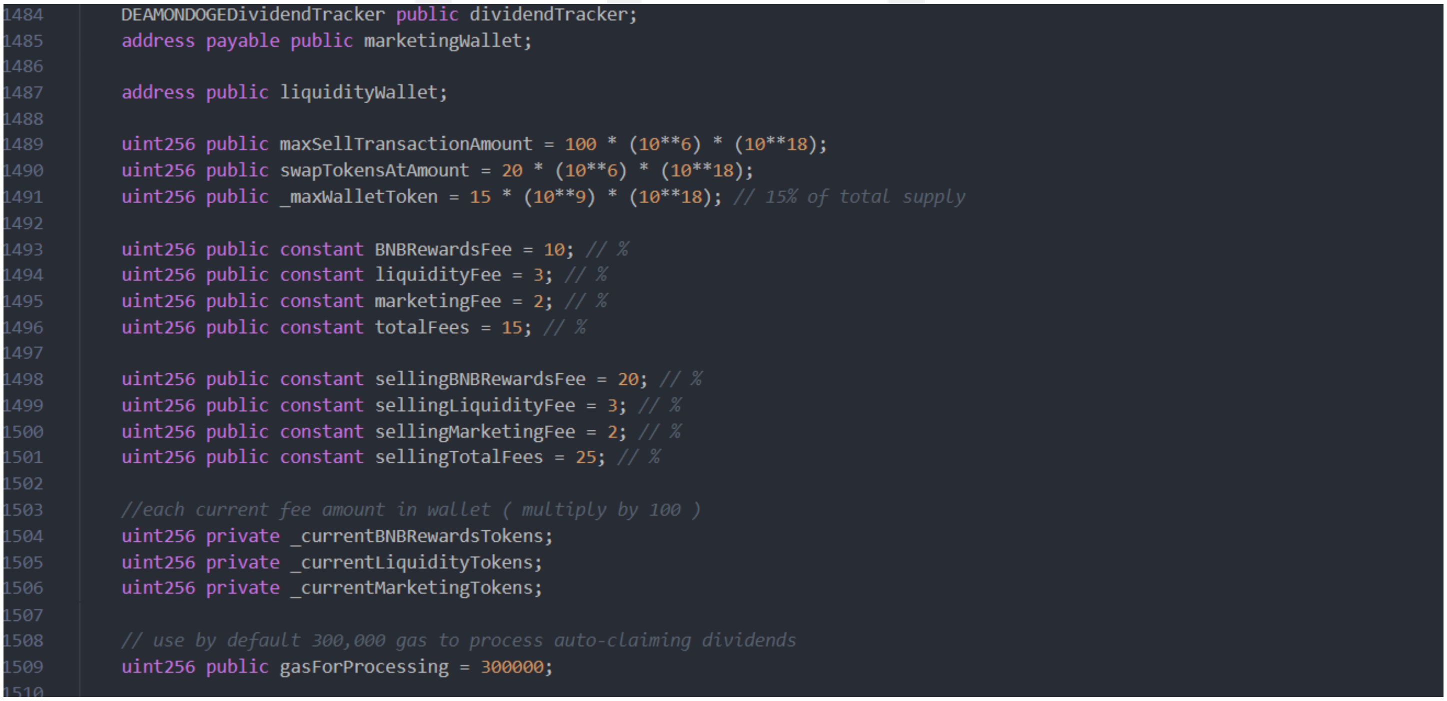
Points To Note

1. The smart contract utilizes the SafeMath to prevent Integer Overflow.



```
contract ERC20 is Context, IERC20, IERC20Metadata {
    using SafeMath for uint256;

    mapping(address => uint256) private _balances;

    mapping(address => mapping(address => uint256)) private _allowances;

    uint256 private _totalSupply;

    string private _name;
    string private _symbol;
```

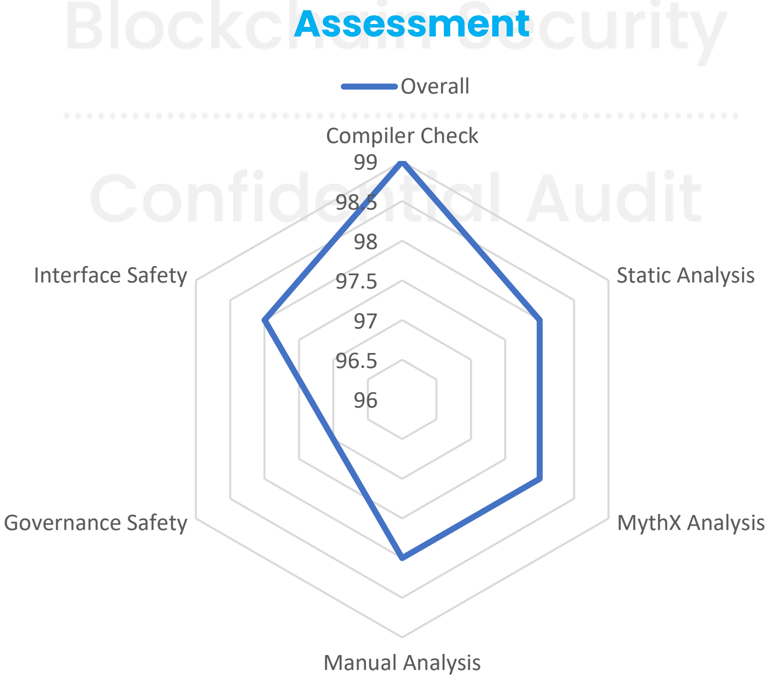
2. The smart contract dictates the buying tax at 15%, and selling tax at 25%



```
    DEAMONDOGEDividendTracker public dividendTracker;
    address payable public marketingWallet;

    address public liquidityWallet;

    uint256 public maxSellTransactionAmount = 100 * (10**6) * (10**18);
    uint256 public swapTokensAtAmount = 20 * (10**6) * (10**18);
    uint256 public _maxWalletToken = 15 * (10**9) * (10**18); // 15% of total supply

    uint256 public constant BNBRewardsFee = 10; // %
    uint256 public constant liquidityFee = 3; // %
    uint256 public constant marketingFee = 2; // %
    uint256 public constant totalFees = 15; // %

    uint256 public constant sellingBNBRewardsFee = 20; // %
    uint256 public constant sellingLiquidityFee = 3; // %
    uint256 public constant sellingMarketingFee = 2; // %
    uint256 public constant sellingTotalFees = 25; // %

    //each current fee amount in wallet ( multiply by 100 )
    uint256 private _currentBNBRewardsTokens;
    uint256 private _currentLiquidityTokens;
    uint256 private _currentMarketingTokens;

    // use by default 300,000 gas to process auto-claiming dividends
    uint256 public gasForProcessing = 300000;
```

Vulnerability	Status
Compiler errors	! Low
Re-entrancy. Race conditions and cross function race conditions (RE)	Passed
Possible delays in data delivery	Passed
Gas optimization	Passed
Integer Underflow and overflow	Passed
Oracle Calls	Passed
Call stack depth attack	Passed
Parity Multisig Bug	Passed
Tx ordering dependency (TO)	Passed
DOS with revert and block gas limit	Passed
Private user data leaks	Passed
Malicious event log	Passed
Safe open zeppelin contract implementation and usage	Passed
The impact of exchange rate on the logic	Passed
Functions that are not used (dead-code)	! Low
Typographical Errors	! Low
Signature Malleability	Passed
Floating Pragma	Passed
Scoping and declarations	Passed

Risk Severity	Status
! Critical	None critical severity issues identified
! High	None high severity issues identified
! Medium	None medium severity issues identified
! Low	Low issues identified (1) *MythX SWC-103. A floating pragma is set.* ***(No Impact)***



Conclusion

InterFi team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks.

Daemon Doge's smart contract source code has LOW RISK SEVERITY.

Daemon Doge has PASSED the InterFi's ECHELON-1 standard smart contract audit.

Auditor's Footnote:

- At the time of the audit, the contract is not deployed on any blockchain. Note, the owner/developer can change/modify contract before deployment. Please proceed with caution.
- Liquidity pair contract's security is not checked due to out of scope. Liquidity locking details NOT provided by the team.
- Project website is not checked due to out of scope. The website hasn't been reviewed for SSL and lighthouse report.
- Project team, and the project's social channels are not checked due to out of scope.

Important Disclaimer

InterFi Network provides contract auditing and project verification services for blockchain projects. The purpose of the audit is to analyse the on-chain smart contract source code, and to provide basic overview of the project. **This report should not be transmitted, disclosed, referred to, or relied upon by any person for any purposes without InterFi's prior written consent.**

InterFi provides the easy-to-understand assessment of the project, and the smart contract (otherwise known as the source code). The audit makes no statements or warranties on the security of the code. It also cannot be considered as an enough assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. While we have used all the data at our disposal to provide the transparent analysis, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. **Be aware that smart contracts deployed on a blockchain aren't resistant from external vulnerability, or a hack. Therefore, InterFi does not guarantee the explicit security of the audited smart contract.**

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.

This report should not be considered as an endorsement or disapproval of any project or team. The information provided on this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. Do conduct your own due diligence and consult your financial advisor before making any investment decisions.

About InterFi Network

InterFi Network provides intelligent blockchain solutions. InterFi is developing an ecosystem that is seamless and responsive. Some of our services: Blockchain Security, Token Launchpad, NFT Marketplace, etc. **InterFi's mission is to interconnect multiple services like Blockchain Security, DeFi, Gaming, and Marketplace under one ecosystem that is seamless, multi-chain compatible, scalable, secure, fast, responsive, and easy-to-use.**

InterFi is built by a decentralized team of UI experts, contributors, engineers, and enthusiasts from all over the world. Our team currently consists of 6+ core team members, and 10+ casual contributors. **InterFi provides manual, static, and automatic smart contract analysis, to ensure that project is checked against known attacks and potential vulnerabilities.**

For more information, visit https://interfi.network

To book an audit, message https://t.me/interfiaudits