AUDIT

Dollarmoon

CoinMooner
Find Next Moonshot coins

TABLE OF CONTENTS

AUDIT SUMMARY

This report was written for dollarmoon (Dmoon) in order to find flaws and vulnerabilities in the Dmoon project's source code, as well as any contract dependencies that weren't part of an officially recognized library.

A comprehensive examination has been performed, utilizing Static Analysis, Manual Review, and Dmoon Deployment techniques. The auditing process pays special attention to the following considerations:

- ❖ Testing the smart contracts against both common and uncommon attack vectors
- ❖ Assessing the codebase to ensure compliance with current best practices and industry standards
- ❖ Ensuring contract logic meets the specifications and intentions of the client
- ❖ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- ❖ Through line-by-line manual review of the entire codebase by industry expert

AUDIT OVERVIEW

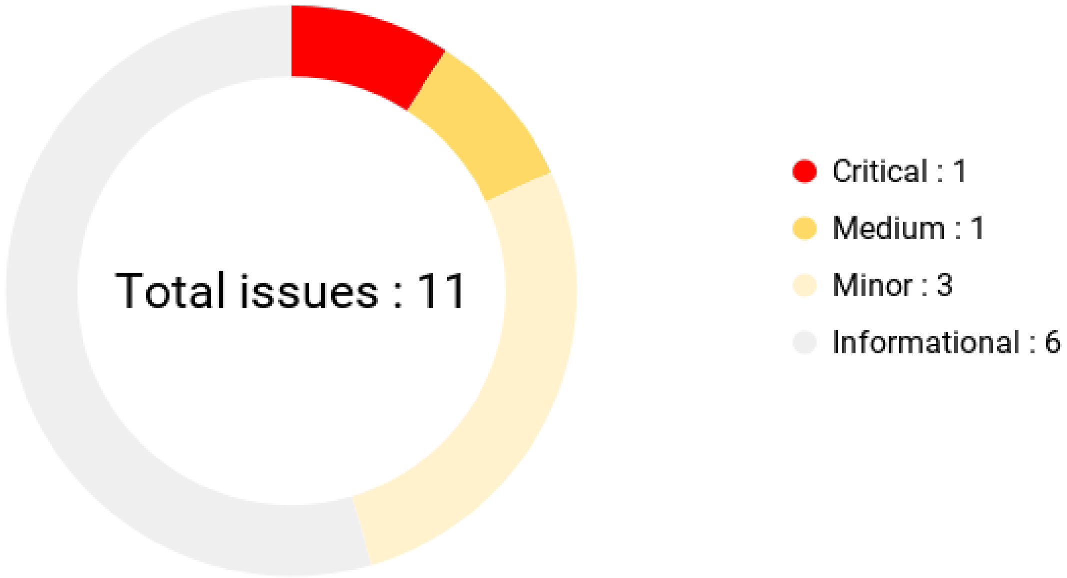
PROJECT SUMMARY

Project name	dollarmoon
Description	People in crypto follow similar goals - financial freedom, success, the best lifestyle… The true definition of the Moon in crypto! Something that seems to be far, but together we can make it happen. Our project will go to the Moon by the well-prepared in details plan, so join our rocket and let's go!
Platform	BNB Smart Chain
Language	Solidity
Codebase	https://bscscan.com/token/0x7d18f3fe6e638fad0adacc5db1a47f871a2c2cc4

FINDINGS SUMMARY

Vulnerability	Total
● Critical	1
● Major	0
● Medium	1
● Minor	3
● Informational	6

AUDIT FINDINGS



Code	Title	Severity
UPDT-1	updateDividendTracker	● Informational
SET-1	setMaxSellTransaction	● Medium
SET-2	setSwapTokensAtAmount	● Minor
UPDT-2	updateMinimumBalanceForDividends	● Minor
EXCL-1	excludeFromFees	● Informational
SET-3	setAutomatedMarketMakerPair	● Informational
UPDT-3	updateLiquidityWallet	● Minor

UPDT-4	updateFee Functions	● Informational
SET-4	setFeeStatus, setTradeFeeStatus	● Informational
TRFR-1	_transfer	● Critical
SWAP-1	swapTokensForEth,swapAndSendMarketingBNB	● Informational

UPDT-1 | updateDividendTracker

Description

The pair address should be excluded here.
_setAutomatedMarketMakerPair function does that for the token, but It’s just extra code. Also, this function should be called in the constructor instead of manually excluding the addresses again.
Extra code means extra gas.

SET-1 | setMaxSellTransaction

Description

There are no thresholds for the `maxSellTransactionAmount` value. The owner can set this variable to zero after the selling is not possible.

SET-2 | setSwapTokensAtAmount

Description

There are no thresholds for the swapTokensAtAmount value. If the owner sets that variable to zero, dividend tokens are worth nothing and they won’t bring any passive income.

UPDT-2 | updateMinimumBalanceForDividends

Description

There are no thresholds for the `newMinimumBalance` parameter. If the owner sets that variable to an absurd value like MaxUInt256 dividend tokens are not gonna work and they won’t bring any passive income.

EXCL-1 | excludeFromFees

Description

```
require(_isExcludedFromFees[account] != excluded, "DollarMoon: Account is already the value of 'excluded'");
```

This piece of code is unnecessary because even though the account is already excluded, It won’t change the outcome.

SET-3 | setAutomatedMarketMakerPair

Description

_setAutomatedMarketMakerPair is a private function but It doesn’t need to be a private function. You can do the same job with only one public owner only function thus gas optimization.

UPDT-3 | updateLiquidityWallet

Description

There are no dead address checks for the newLiquidityWallet variable. If it’s set to a null or dead address, liquidity fees will get burnt.

UPDT-4 | updateFee Functions

Description

The fee variables are important variables so after they are changed there should be an event emitted.

SET-4 | setFeeStatus, setTradeFeeStatus

Description

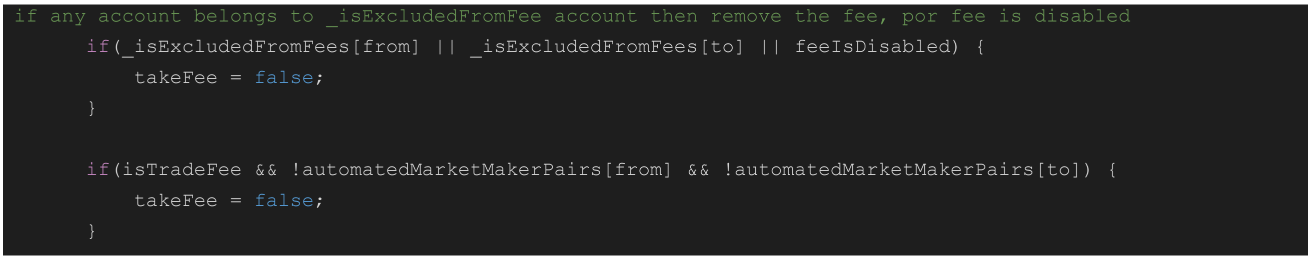
The ‘require’ codes are unnecessary because they won’t change anything If the boolean values are false. Gas optimization.

TRFR-1 | _transfer

Description

The code block is hard to understand for a normal person. There are a lot of unnecessary code lines, and the code block needs to be prettified and cleared for unnecessary code lines that use unnecessary gas.

Those lines can be combined into one.



```
if any account belongs to _isExcludedFromFee account then remove the fee, por fee is disabled
      if(_isExcludedFromFees[from] || _isExcludedFromFees[to] || feeIsDisabled) {
          takeFee = false;
      }

      if(isTradeFee && !automatedMarketMakerPairs[from] && !automatedMarketMakerPairs[to]) {
          takeFee = false;
      }
```

If the setBalance fails during the transfer, even though the tokens do transfer to another wallet; the dividend token’s won’t update and the recipient or the sender’s dividend tracker might not update.

If the sender’s setBalance fails even though he/she sent all his tokens to the recipient, the dividend tokens will stay the same and he/she might receive free distribution WBNB’s.

SWAP-1| swapTokensForEth, swapAndSendMarketingBNB

Description

They are the same functions with different addresses. They should be combined into one function for gas optimization.

Tax Info

The total trading fees of the token contract.

- Buy Fee: 12%
- Sell Fee: 12%

Comments

The token contract takes fees from each trade; liquidity fee, burning fee, distribution fee, marketing fee and burn fee. Liquidity fee gets added to the Pancakeswap liquidity of the token, burning fee gets burnt, marketing fee goes directly to the marketing wallet. The distribution fee gets swapped into WBNB after that it is distributed with distribution tokens.

DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement.

This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement.

This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CoinMooner's prior written consent.This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts CoinMooner to perform a security assessment.

This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with

any particular project.
This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk.
CoinMooner's position is that each company and individual are responsible for their own due diligence and continuous security. CoinMooner's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or fun.