

Smart contract security audit

Babydot

v.1.0



No part of this publication, in whole or in part, may be reproduced, copied, transferred or any other right reserved to its copyright a CTDSec, including photocopying and all other copying, any transfer or transmission using any network or other means of communication, in any form or by any means such as any information storage, transmission or retrieval system, without prior written permission.

Table of Contents

1.0 Introduction	3
1.1 Project engagement	3
1.2 Disclaimer	3
2.0 Coverage	4
2.1 Target Code and Revision	4
2.2 Attacks made to the contract	5
3.0 Security Issues	7
3.1 High severity issues [0]	7
3.2 Medium severity issues [0]	7
3.3 Low severity issues [1]	7
4.0 Owner Privileges	8
5.0 Summary of the audit	11

1.0 Introduction

1.1 Project engagement

During July of 2021, Babydot engaged CTDSec to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. Babydot provided CTDSec with access to their code repository and whitepaper.

1.2 Disclaimer

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that's able to withstand the network's fast-paced and rapidly changing environment, we at CTDSec recommend that Babydot team put in place a bug bounty program to encourage further and active analysis of the smart contract.

2.0 Coverage

2.1 Target Code and Revision

For this audit, we performed research, investigation, and review of the Babydot contract followed by issue reporting, along with mitigation and remediation instructions outlined in this report. The following code files are considered in-scope for the review:

Source:

<https://bscscan.com/address/0x08bd7f9849f8eec12fd78c9fed6ba4e47269e3d5#code>

2.2 Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices.

No	Issue description.	Checking status
1	Compiler warnings.	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. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	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
22	Overpowered functions / Owner privileges	PASSED

3.0 Security Issues

3.1 High severity issues [0]

No high severity issues found.

3.2 Medium severity issues [0]

No medium severity issues found.

3.3 Low severity issues [1]

1. Out of gas Issue:

Issue:

The function **SetupAddOrRemoveWhitelist()** uses the loop to add or remove whitelist addresses. Function will be aborted with OUT_OF_GAS exception if there will be a long addresses list.

Recommendation:

Check that the array length is not too big.

4.0 Owner Privileges

- Team can change diamondPaws settings.
- Team can enable taxfree compound calls.
- Team can set autoLP threshold.
- Team can exclude and include in staking.
- Team can withdraw marketing balance.
- Team can disable auto call of `_swapContractToken` function.
- Team can change sell lock time.
- Team can change taxes.
- Team can change marketing share (percentage of BNB that goes to marketing).
- Team can manually call `_swapContractToken` function.
- Team can exclude from fees and sell lock.
- Team can disable sellLock.
- Team can withdraw team part of the contract balance.
- Team can change target balance and sell limits.
- Team can enable trading.
- Team can call `TeamLimitLiquidityReleaseTo20Percent()` to set only 20% withdraw available.
- Team can increase `_liquidityUnlockTime`.
- Team can withdraw liquidity to team wallet if it is not locked.
- Team can remove liquidity.
- Team can withdraw contract BNBs if it is not locked.
- Team can withdraw BEP20 tokens from the contract.

5.0 Summary of the audit

Liquidity locking details aren't provided. Smart contracts contain low issues and it's safe to deploy.