HALBORN

Trust Wallet - Barz

Executive Summary

Prepared by: **Halborn**

Date of Engagement: **May 22nd, 2023 - June 12th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE
0.1	Document Creation	06/10/2023
0.2	Document Edits	06/11/2023
0.3	Draft Review	06/12/2023
0.4	Draft Review	06/12/2023
0.5	Draft Review	06/12/2023
1.0	Remediation Plan	06/20/2023
1.1	Remediation Plan Review	06/20/2023
1.2	Remediation Plan Review	06/20/2023
1.3	Remediation Plan Review	06/20/2023
1.3	Remediation Plan Review	06/21/2023

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Trust Wallet engaged Halborn to conduct a security assessment on their smart contracts beginning on May 22nd, 2023 and ending on June 12th, 2023. The security assessment was scoped to the smart contracts provided to the Halborn team.

The Barz project is a wallet project that has a diamond proxy structure with useful facets, enabling mass adoption of Web3 with security, modularity, and upgradability.

1.2 ASSESSMENT SUMMARY

The team at Halborn was provided three weeks for the engagement and assigned a full-time security engineer to verify the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this assessment is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some security risks that were successfully addressed by the Trust Wallet team. The main ones were the following:

- GUARDIAN THRESHOLD CAN BE BYPASSED WITH REPETITIVE DATA
- FRONT-RUNNING RESTRICTION INITIALIZATION CAN LOCK THE WALLET
- BROKEN UNLOCK FUNCTIONALITY
- WHITELISTING AND BLACKLISTING ARE IMPOSSIBLE IN NO-GUARDIAN CONDITION
- INSUFFICIENT METHOD SELECTOR CONTROL CAN LEAD TO RESTRICTION CHECK BYPASS

EXECUTIVE OVERVIEW

Halborn's findings, descriptions and remediations have been redacted at the request of Trust Wallet.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy regarding the scope of the smart contract assessment. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of smart contracts and can quickly identify items that don't follow security best practices. The following phases and associated tools were used throughout the term of the assessment:

- Research into architecture and purpose.
- Smart Contract manual code review and walkthrough.
- Graphing out functionality and contract logic/connectivity/functions(solgraph)
- Manual Assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes.
- Dynamic Analysis (foundry)
- Static Analysis(slither, MythX)

2. RISK METHODOLOGY

Every vulnerability and issue observed by Halborn is ranked based on **two sets** of **Metrics** and a **Severity Coefficient**. This system is inspired by the industry standard Common Vulnerability Scoring System.

The two **Metric sets** are: **Exploitability** and **Impact**. **Exploitability** captures the ease and technical means by which vulnerabilities can be exploited and **Impact** describes the consequences of a successful exploit.

The **Severity Coefficients** is designed to further refine the accuracy of the ranking with two factors: **Reversibility** and **Scope**. These capture the impact of the vulnerability on the environment as well as the number of users and smart contracts affected.

The final score is a value between 0-10 rounded up to 1 decimal place and 10 corresponding to the highest security risk. This provides an objective and accurate rating of the severity of security vulnerabilities in smart contracts.

The system is designed to assist in identifying and prioritizing vulnerabilities based on their level of risk to address the most critical issues in a timely manner.

2.1 EXPLOITABILITY

Attack Origin (AO):

Captures whether the attack requires compromising a specific account.

Attack Cost (AC):

Captures the cost of exploiting the vulnerability incurred by the attacker relative to sending a single transaction on the relevant blockchain. Includes but is not limited to financial and computational cost.

Attack Complexity (AX):

Describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. Includes but is not limited to macro situation, available third-party liquidity and regulatory challenges.

Metrics:

Exploitability Metric (m_E)	Metric Value	Numerical Value
Attack Origin (AO)	Arbitrary (AO:A) Specific (AO:S)	1 0.2
Attack Cost (AC)	Low (AC:L) Medium (AC:M) High (AC:H)	1 0.67 0.33
Attack Complexity (AX)	Low (AX:L) Medium (AX:M) High (AX:H)	1 0.67 0.33

Exploitability E is calculated using the following formula:

$$E = \prod m_e$$

2.2 IMPACT

Confidentiality (C):

Measures the impact to the confidentiality of the information resources managed by the contract due to a successfully exploited vulnerability. Confidentiality refers to limiting access to authorized users only.

Integrity (I):

Measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of data stored and/or processed on-chain. Integrity impact directly affecting Deposit or Yield records is excluded.

Availability (A):

Measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. This metric refers to smart contract features and functionality, not state. Availability impact directly affecting Deposit or Yield is excluded.

Deposit (D):

Measures the impact to the deposits made to the contract by either users or owners.

Yield (Y):

Measures the impact to the yield generated by the contract for either users or owners.

Metrics:

Impact Metric (m_I)	Metric Value	Numerical Value
Confidentiality (C)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Integrity (I)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Availability (A)	None (A:N)	0
	Low (A:L)	0.25
	Medium (A:M)	0.5
	High (A:H)	0.75
	Critical	1
Deposit (D)	None (D:N)	0
	Low (D:L)	0.25
	Medium (D:M)	0.5
	High (D:H)	0.75
	Critical (D:C)	1
Yield (Y)	None (Y:N)	0
	Low (Y:L)	0.25
	Medium: (Y:M)	0.5
	High: (Y:H)	0.75
	Critical (Y:H)	1

Impact I is calculated using the following formula:

$$I = max(m_I) + \frac{\sum m_I - max(m_I)}{4}$$

2.3 SEVERITY COEFFICIENT

Reversibility (R):

Describes the share of the exploited vulnerability effects that can be reversed. For upgradeable contracts, assume the contract private key is available.

Scope (S):

Captures whether a vulnerability in one vulnerable contract impacts resources in other contracts.

Coefficient (C)	Coefficient Value	Numerical Value
Reversibility (r)	None (R:N) Partial (R:P) Full (R:F)	1 0.5 0.25
Scope (s)	Changed (S:C) Unchanged (S:U)	1.25 1

Severity Coefficient C is obtained by the following product:

$$C = rs$$

The Vulnerability Severity Score S is obtained by:

$$S = min(10, EIC * 10)$$

The score is rounded up to 1 decimal places.

Severity	Score Value Range
Critical	9 - 10
High	7 - 8.9
Medium	4.5 - 6.9
Low	2 - 4.4
Informational	0 - 1.9

2.4 SCOPE & FINDINGS OVERVIEW

The Trust Wallet team shared the project source code in a compressed file.

- Barz Smart Contracts
  - First Commit ID: 03d0019f4616712dcdca6e291af4c809358392cf
  - Final Commit ID: a97080f05e029eab93b62cf0cbd8ce71417fc0bd
  - In-Scope:
    - AccountFacet.sol
    - AccountRecoveryFacet.sol
    - GuardianFacet.sol
    - LockFacet.sol
    - RestrictionsFacet.sol
    - SignatureMigrationFacet.sol
    - Barz.sol
    - BarzFactory.sol
    - LibAppStorage.sol
    - LibDiamond.sol
    - LibFacetStorage.sol
    - LibRecoverSpender.sol
    - TokenReceiverFacet.sol
    - DefaultFallbackHandler.sol
    - LibUtils.sol
    - LibLoupe.sol
    - DefaultLibDiamond.sol
    - Modifiers.sol
    - infrastructure/*.sol
    - base/*.sol
    - verification/*.sol
    - restrictions/*.sol
    - interfaces/*.sol
  - Out-of-Scope:
    - contracts/aa-4337/*.sol
    - contracts/test/*.sol

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) GUARDIAN THRESHOLD CAN BE BYPASSED WITH REPETITIVE DATA	High (8.4)	SOLVED - 06/18/2023
(HAL-02) FRONT-RUNNING RESTRICTION INITIALIZATION CAN LOCK THE WALLET	High (8.0)	SOLVED - 06/18/2023
(HAL-03) BROKEN UNLOCK FUNCTIONALITY	Medium (5.0)	SOLVED - 06/09/2023
(HAL-04) WHITELISTING AND BLACKLISTING ARE IMPOSSIBLE IN NO-GUARDIAN CONDITION	Medium (5.0)	SOLVED - 06/09/2023
(HAL-05) INSUFFICIENT METHOD SELECTOR CONTROL CAN LEAD TO RESTRICTION CHECK BYPASS	Medium (5.0)	SOLVED - 06/18/2023
(HAL-06) LACK OF ZERO ADDRESS CHECKS	Low (2.5)	SOLVED - 06/18/2023
(HAL-07) FLOATING PRAGMA	Informational (0.0)	SOLVED - 06/18/2023
(HAL-08) MISSING FUNCTIONS IN INTERFACE CONTRACTS	Informational (0.0)	SOLVED - 06/18/2023
(HAL-09) UNUSED IMPORTS, FUNCTIONS AND EVENTS	Informational (0.0)	SOLVED - 06/18/2023
(HAL-10) EMITTED EVENT MIGHT BE CONFUSING FOR USERS	Informational (0.0)	SOLVED - 06/18/2023

THANK YOU FOR CHOOSING

// HALBORN