// HALBORN

C3 - PyTeal

Executive Summary

Prepared by: **Halborn**

Date of Engagement: **July 20th, 2023 - August 9th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE
0.1	Document Creation	07/20/2023
0.2	Document Updates	08/03/2023
0.3	Draft Review	08/09/2023
1.0	Remediation Plan	08/30/2023
1.1	Remediation Plan Review	08/31/2023
2.0	Executive Summary	09/27/2023

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
Gokberk Gulgun	Halborn	Gokberk Gulgun@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

C3 engaged Halborn to conduct a security assessment on their PyTeal smart contracts beginning on July 20th, 2023 and ending on August 9th, 2023. The security assessment was scoped to the vesting module provided to the Halborn team.

1.2 ASSESSMENT SUMMARY

The team at Halborn was provided one month for the engagement and assigned two full-time security engineers to assessment the security of the merge requests. The security engineers are blockchain and smart-contract security experts with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this assessment is to:

- Ensure that the **smart contracts** operates as intended.
- Identify potential security issues with the PyTeal smart contract.

In summary, Halborn identified some improvements to reduce the likelihood and impact of risks that were mostly addressed by the C3 team.

The outcome of this Algorand Smart Contract Security Assessment engagement is provided as a detailed technical report that provides the Smart Contract owners a full description of the vulnerabilities identified, the associated risk rating for each vulnerability, and detailed recommendations that will resolve the identified technical issue.

1.3 TEST APPROACH & METHODOLOGY

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

- Research into architecture and purpose.
- Static Analysis of security for scoped repository, and imported functions. (e.g., semgrep)
- Manual Assessment for discovering security vulnerabilities on codebase.
- Ensuring correctness of the codebase.
- Dynamic Analysis on files and smart contracts related to the **C3**.

An assessment was conducted on the provided PyTeal code to identify any weaknesses, vulnerabilities, and non-compliance to Algorand best practices.

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)	1
	Specific (AO:S)	0.2
Attack Cost (AC)	Low (AC:L)	1
	Medium (AC:M)	0.67
	High (AC:H)	0.33
Attack Complexity (AX)	Low (AX:L)	1
	Medium (AX:M)	0.67
	High (AX:H)	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)	1
	Partial (R:P)	0.5
	Full (R:F)	0.25
Scope (s)	Changed (S:C)	1.25
	Unchanged (S:U)	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

This review was scoped to the on the **C3 contracts-unified** repository.

Halborn's findings, descriptions and remediations have been redacted at the request of C3.

3. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	2	3	3	10

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) LACK OF OWNER AND CREATOR FIELD VALIDATION IN ORDERDATA LEADING TO PHISHING ATTACKS	High	SOLVED - 08/31/2023
(HAL-02) LACK OF VAA CHECKS LEADS TO FAKE DEPOSITS THROUGH THE WORMHOLE	High	SOLVED - 08/31/2023
(HAL-03) UNBOUNDED WITHDRAWAL FEE IN SMART CONTRACT LEADING TO POTENTIAL EXPLOITATION	Medium	SOLVED - 08/31/2023
(HAL-04) MISSING CHECKS FOR GROUP SIZE AND GROUP INDEX	Medium	SOLVED - 08/31/2023
(HAL-05) ORDER SETTLEMENT DOES NOT CHECK HEALTH OF ACCOUNTS	Medium	SOLVED - 08/31/2023
(HAL-06) LACK OF INPUT VALIDATION FOR COMPOUND TYPES IN PYTEAL’S ROUTER CLASS	Low	SOLVED - 08/31/2023
(HAL-07) LACK OF PAUSE FUNCTIONALITY	Low	SOLVED - 08/31/2023
(HAL-08) LACK OF VALIDATION IN PORTAL TRANSFER FUNCTION REGARDING THE ORIGIN FROM WORMHOLE CONTRACT	Low	FUTURE RELEASE
(HAL-09) ARC-04 SMART CONTRACT WITH ROUTER CLASS WITHOUT PRAGMA OR PRAGMA	Informational	ACKNOWLEDGED
(HAL-10) USAGE OF LOG FUNCTION	Informational	SOLVED - 08/31/2023
(HAL-11) ENSURE BUDGET IS NOT UTILIZED	Informational	ACKNOWLEDGED
(HAL-12) DEPOSIT FUNCTION IS MISSING VALIDATESENDER CHECK	Informational	SOLVED - 08/31/2023
(HAL-13) USE OF SETTLE ORDER ID IN ADD ORDER FUNCTION	Informational	ACKNOWLEDGED
(HAL-14) LACK OF EXPLICIT CHECK FOR MAX BORROW IN THE WITHDRAW FUNCTION	Informational	ACKNOWLEDGED

(HAL-15) INSUFFICIENT VALIDATION OF APPLICATION INITIALIZATION ARGUMENTS	Informational	SOLVED - 08/31/2023
(HAL-16) ABSENCE OF MINIMUM DEPOSIT REQUIREMENT IN DEPOSIT FUNCTION	Informational	ACKNOWLEDGED
(HAL-17) ADDING VALIDATION CHECKS TO THE add order FUNCTION	Informational	ACKNOWLEDGED
(HAL-18) USE OF ASSERT OVER POP FOR BOX DELETION VERIFICATION	Informational	ACKNOWLEDGED

THANK YOU FOR CHOOSING

HALBORN