// HALBORN

Bracket.fi - Passage

Executive Summary

Prepared by: **Halborn**

Date of Engagement: **March 20th, 2023 - April 15th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY 2
CONTACTS 3
1 EXECUTIVE OVERVIEW 4
1.1 INTRODUCTION 5
1.2 ASSESSMENT SUMMARY 5
1.3 TEST APPROACH & METHODOLOGY 6
2 RISK METHODOLOGY 8
2.1 EXPLOITABILITY 9
2.2 IMPACT 10
2.3 SEVERITY COEFFICIENT 12
2.4 SCOPE 14

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE
0.1	Document Creation	04/13/2023
0.2	Document Updates	04/14/2023
0.3	Document Updates	04/15/2023
0.4	Draft Version	04/16/2023
0.4	Draft Review	04/17/2023
0.5	Draft Review	04/17/2023
0.6	Draft Review	04/18/2023
1.0	Remediation Plan	05/29/2023
1.1	Remediation Plan Update	05/30/2023
1.2	Remediation Plan Review	06/06/2023
1.3	Remediation Plan Review	06/06/2023
1.4	Remediation Plan Review	06/07/2023

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
Piotr Cielas	Halborn	Piotr.Cielas@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Passage is a decentralized application that allows users to invest in favor or against volatility in an asset with an order book-based model. Users can bet whether the price of an asset stays in a pre-defined range or goes out of it in a pre-defined time span.

Bracket.fi engaged Halborn to conduct a security assessment on their smart contracts beginning on March 20th, 2023 and ending on April 15th, 2023 . The security assessment was scoped to the new Passage product from Bracket.fi.

Commit hashes and further details can be found in the Scope section of this report.

1.2 ASSESSMENT SUMMARY

The team at Halborn was provided 3 weeks for the engagement and assigned 1 full-time security engineer to verify the security of the smart contracts in scope. The assessment took a total of 4 weeks, with a one-week pause in the middle of the assessment.

The security engineer in charge of the assessment is a blockchain and smart contract security expert with advanced penetration testing and smart contract hacking skills, and deep knowledge of multiple blockchain protocols.

The purpose of the assessments is to:

- Identify potential security issues within the smart contracts.
- Ensure that smart contract functionality operates as intended.

In summary, Halborn identified 6 critical, 3 medium and 6 informational

issues that were successfully addressed by the Bracket.fi team. The main ones were the following:

- Modify the liquidity pool LP token calculation formula to prevent an attacker from stealing other users' funds.
- Include the nonReentrant modifier or modify the deposit function logic to prevent a possible reentrancy attack when minting ERC1155 lp tokens.
- Ensure pointer, counter and amount of variables are properly tracked to prevent funds from being locked.

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

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 this assessment. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the code and can quickly identify items that do not follow the security best practices. The following phases and associated tools were used during 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

- Manual testing by custom scripts
- Scanning of solidity files for vulnerabilities, security hot-spots or bugs. (MythX)
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Testnet deployment (Brownie, Remix IDE, Foundry)

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

Code repositories:

1. Passage

- Repository: bracketx-halborn
- Commit ID: c66e52967d9ac4b7d54fde581924093c2ca62902
- Smart contracts in scope:
    1. Passage.sol (src/Passage.sol)
    2. PassageLib.sol (src/PassageLib.sol)
    3. BPLP.sol (src/BPLP.sol)
    4. BPLPLib.sol (src/BPLPLib.sol)
    5. BktMath.sol (src/BktMath.sol)
    6. PNFT.sol (src/BktMath.sol)
- Fixes Commit IDs:
    - ca7bd2b1fcb48baf7fc5151762772e32335667b4.
    - 70ddfa16f0edaa8202d2c3837877d0b28e430f4e
    - 05907076d05cd4486e179cfa9fee03a34b88e755
    - 78f58a337fa0ad0b318960ea10e3846bd6082e04
    - 3478967b0b801cf2217483ffa2109461d6b5a21c
    - 3329572c7c04cff0a00ee5d6f9c04ccbc2a52c43

Out-of-scope:
- Third-party libraries and dependencies
- Economic attacks

THANK YOU FOR CHOOSING

// HALBORN