Security Assessment Report

Monaco Protocol v0.15.5

December 11, 2024

Summary

The Sec3 team (formerly Soteria) was engaged to conduct a thorough security analysis of the Monaco Protocol v0.15.5.

The artifact of the audit was the source code of the following programs, excluding tests, in a https://github.com/MonacoProtocol/protocol/.

The initial audit focused on the following versions and revealed 1 issues or questions.

program	type	commit
monaco_protocol v0.15.5	Solana	e22a6b17c6cf06687d66998e1d0b221df7e73dc0

The post-audit review was conducted on the following version to check if the reported issues had been addressed.

program	type	commit
monaco_protocol v0.15.5	solana	96d4d7976601f17d0a3bac6e801f92f0d66a4a50

This report provides a detailed description of the findings and their respective resolutions.

Table of Contents

Result Overview

Issue	Impact	Status
MONACO_PROTOCOL V0.15.5		
[L-01] Inconsistent "voided_stake" tracking	Low	Resolved

Findings in Detail

MONACO_PROTOCOL V0.15.5

[L-01] Inconsistent "voided_stake" tracking

With the newly added "`void_stake_unmatched_by()`", the "`order.voided_stake`" may become inconsistent, even though the order status is correctly set.

```
/* programs/monaco_protocol/src/state/order_account.rs */
064 | pub fn void_stake_unmatched(&mut self) {
065 |     self.voided_stake = self.stake_unmatched;
066 |     self.stake_unmatched = 0_u64;
067 |     if self.order_status == OrderStatus::Open {
068 |         self.order_status = OrderStatus::Cancelled;
069 |     }
070 | }
071 |
072 | pub fn void_stake_unmatched_by(&mut self, stake_to_void: u64) -> Result<()> {
073 |     self.voided_stake = self
074 |         .voided_stake
075 |         .checked_add(stake_to_void)
076 |         .ok_or(CoreError::ArithmeticError)?;
077 |     self.stake_unmatched = self
078 |         .stake_unmatched
079 |         .checked_sub(stake_to_void)
080 |         .ok_or(CoreError::ArithmeticError)?;
081 |
082 |     if self.stake == self.voided_stake && self.order_status == OrderStatus::Open {
083 |         self.order_status = OrderStatus::Cancelled;
084 |     }
085 |
086 |     Ok(())
087 | }
```

Consider the following example, assuming the initial state is as follows:

```
order.stake = 10
order.voided_stake = 0
order.unmatched = 10
order.order_status = Open
```

Perform a partial cancellation by calling "`void_stake_unmatched_by(2)`" (via "`cancel_order()`").

```
order.stake = 10
order.voided_stake = 2
order.unmatched = 8
order.order_status = Open
```

Perform a full cancellation by calling "`void_stake_unmatched()`" via "`settle_order`" or "`cancel_preplay_order_post_event_start`".

```
order.stake = 10
order.voided_stake = 8
order.unmatched = 0
order.order_status = Cancelled
```

Although the order status is "`Cancelled`" and it is not matched with others, the "`voided_stake`" incorrectly becomes "`8`" instead of "`10`".

Resolution

Fixed by commit 96d4d79.

Appendix: Methodology and Scope of Work

Assisted by the Sec3 Scanner developed in-house, the manual audit particularly focused on the following work items:

- Check common security issues.
- Check program logic implementation against available design specifications.
- Check poor coding practices and unsafe behavior.
- The soundness of the economics design and algorithm is out of scope of this work

DISCLAIMER

The instance report ("Report") was prepared pursuant to an agreement between Coderrect Inc. d/b/a Sec3 (the "Company") and BetDEX Labs (the "Client''). This Report solely includes the results of a technical assessment of a specific build and/or version of the Client's code specified in the Report ("Assessed Code") by the Company. The sole purpose of the Report is to provide the Client with the results of the technical assessment of the Assessed Code. The Report does not apply to any other version and/or build of the Assessed Code. Regardless of the contents of the Report, the Report does not (and should not be interpreted to) provide any warranty, representation or covenant that the Assessed Code: (i) is error and/or bug free, (ii) has no security vulnerabilities, and/or (iii) does not infringe any third-party rights. Moreover, the Report is not, and should not be considered, an endorsement by the Company of the Assessed Code and/or of the Client. Finally, the Report should not be considered investment advice or a recommendation to invest in the Assessed Code and/or the Client.

This Report is considered null and void if the Report (or any portion thereof) is altered in any manner.

ABOUT

The Sec3 audit team comprises a group of computer science professors, researchers, and industry veterans with extensive experience in smart contract security, program analysis, testing, and formal verification. We are also building automated security tools that incorporate static analysis, penetration testing, and formal verification.

At Sec3, we identify and eliminate security vulnerabilities through the most rigorous process and aided by the most advanced analysis tools.

For more information, check out our website and follow us on twitter.