Security Assessment Report

Kamino Vault

April 3, 2025

Summary

The Sec3 team (formerly Soteria) was engaged to conduct a thorough security analysis of the Kamino Vault smart contracts, starting on March 21, 2025.

The artifact of the audit was the source code of the following programs, excluding tests, in a private repository.

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

#	program	type	commit
P1	Kamino Vault	Solana	`e3619fec408c5427ca68cd3dede72867a1f1f4b2`

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

Table of Contents

Result Overview

Issue	Impact	Status
KAMINO VAULT		
[L-01] Potential panic due to insufficient kmsg length	Low	Resolved

Findings in Detail

KAMINO VAULT

[L-01] Potential panic due to insufficient kmsg length

The current implementation uses a custom `kmsg` macro for log output. Internally, `kmsg` allocates a buffer whose size is determined by the length of the format string. For format strings of up to 50 characters, it allocates a fixed 150-byte buffer.

```
/* programs/kamino-vault/src/utils/macros.rs */
138 | /// Log a formatted message with automatic capacity estimation
139 | /// Uses arrform! and msg! together for efficient logging
140 | /// Capacity is conservatively estimated based on format string length
141 | #[macro_export]
142 | macro_rules! kmsg {
166 |     // For formats with arguments
167 |     ($fmt:expr, $($arg:expr),+) => {{
168 |         // Choose capacity tier based on format string length
169 |         // This is very conservative to avoid overflows
170 |         match $fmt.len() {
171 |             0..=50 => {
172 |                 let formatted = $crate::arrform!{150, $fmt, $($arg),+};
173 |                 anchor_lang::prelude::msg!("{}", formatted.as_str());
174 |             },
175 |             51..=100 => {
176 |                 let formatted = $crate::arrform!{300, $fmt, $($arg),+};
177 |                 anchor_lang::prelude::msg!("{}", formatted.as_str());
178 |             },
179 |             101..=200 => {
180 |                 let formatted = $crate::arrform!{600, $fmt, $($arg),+};
181 |                 anchor_lang::prelude::msg!("{}", formatted.as_str());
182 |             },
183 |             _ => {
184 |                 let formatted = $crate::arrform!{1200, $fmt, $($arg),+};
185 |                 anchor_lang::prelude::msg!("{}", formatted.as_str());
186 |             }
187 |         }
188 |     }};
189 | }
```

However, in the invocation shown below, the combination of format string and parameters can produce a formatted message of up to 153 bytes in extreme cases, exceeding the preallocated buffer and causing a runtime panic.

```
/* programs/kamino-vault/src/state.rs */
359 | crate::kmsg!(
360 |     "Reserve {}: {}/{} target {} of total {}",
361 |     allocation.reserve,
362 |     allocation.target_allocation_weight,
363 |     total_weight,
364 |     token_target_allocation.to_floor::<u64>(),
365 |     total_tokens.to_floor::<u64>()
366 | );
```

It is recommended to increase the preallocated buffer size for each format-string length category or use the `kmsg_sized` variant in cases where a large number of formatting parameters is expected.

Resolution

This issue has been fixed by `d6cc3ce`.

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 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.