Accretion

SECURITY ASSESSMENT

Provided by Accretion Labs Pte Ltd. for MRGN
June 19, 2025
A25MRG2

MRGN

AUDITORS

Role	Name
Lead Auditor	Robert Reith (robert@accretion.xyz)
Auditor	Mahdi Rostami (mahdi@accretion.xyz)
Auditor	Niklas Breitfeld (niklas@accretion.xyz)

CLIENT

MRGN (https://marginfi.com) is one of the largest lending protocols in the Solana ecosystem. They engaged Accretion to perform a security assessment of a new integration for Marginfi, adding support for Kamino lending deposits. Using this feature, users can use Kamino deposits as collateral within Marginfi.

ENGAGEMENT SCOPE

Marginfi Kamino Integration
Link: https://github.com/mrgnlabs/marginfi-v2-internal/pull/109
Commit: 61dd5fba4ffaf11308feeccbccd6e97f3cdb1dbc
ProgramID: MFv2hWf31Z9kbCa1snEPYctwafyhdvnV7FZnsebVacA

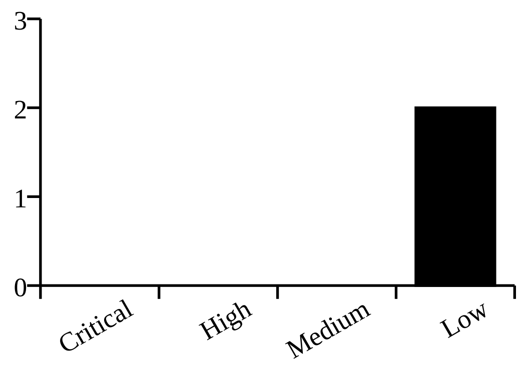
ENGAGEMENT TIMELINE

01 Jun	Project Kickoff Initial planning and scope definition
01 Jun	Assessment Begins Security review and testing phase
17 Jun	Review Fixes Security recommendations are given and implemented
19 Jun	Project Completion Report delivery and on-chain confirmation

ASSESSMENT

The security assessment of the Kamino integration within Marginfi revealed only 3 minor issues, with no critical, high, or medium severity issues found. The identified issues have been addressed by the Marginfi team. The integration is now considered secure for production use.

SEVERITY DISTRIBUTION



AUDITED CODE

Program 1
ProgramID: MFv2hWf31Z9kbCa1snEPYctwafyhdvnV7FZnsebVacA
Repository: https://github.com/mrgnlabs/marginfi-v2-internal/pull/109
Commit: 7bbdf9b67e1bcfce24002c8178aee3130fa17488

ISSUES SUMMARY

ID	TITLE	**SEVERITY**	**STATUS**
ACC-L1	Possible referrer frontrunning in `init_obligation`	low	fixed
ACC-L2	Incorrect Token Transfer Amount in Kamino Deposit	low	accepted
ACC-I1	Missing Associated Token Account Validation in `KaminoHarvestReward`	info	fixed

DETAILED ISSUES

ID	ACC-L1
Title	Possible referrer frontrunning in `init_obligation`
Severity	low
Status	fixed

Description

We found that because the `init_obligation` instruction is permissionless and accepts an arbitrary and optional `referrer_user_metadata` account, a third party may try to wait for any `add_pool` event just to then call the `init_obligation` instruction with their own account as referrer to create the obligation for this bank. This would result in a potential official kamino bank with an external referrer instead of a marginfi internal referrer.

Location

https://github.com/mrgnlabs/marginfi-v2-internal/blob/61dd5fba4ffaf11308feeccbccd6e97f3cdb1dbc/programs/marginfi/src/instructions/kamino/init_obligation.rs#L115-L118

Relevant Code

```
/// init_obligation.rs L115-L118
    /// We may pass in a mfi controlled account as the referrer
    /// CHECK: validated by the Kamino program, generally unrestricted.
    #[account(mut)]
    pub referrer_user_metadata: Option<UncheckedAccount<'info>>,
```

Mitigation Suggestion

We recommend hardcoding a marginfi referrer account, or making the `init_obligation` instruction permissioned to the group admin.

Remediation

Fixed in commit `76682cece52516644abe2b2496a2fc13efa5e4d6`.

ID	ACC-L2
Title	Incorrect Token Transfer Amount in Kamino Deposit
Severity	low
Status	accepted

Description

We found a discrepancy between Klend and Kamino deposit implementations. In Klend, the function calculates a new liquidity deposit amount and transfers exactly that amount from the user [link](https://github.com/Kamino-Finance/klend/blob/812eb90c6e7da9272494f0660f44897ba5eb81d9/programs/klend/src/handlers/handler_deposit_reserve_liquidity_and_obligation_collateral.rs#L92). However, in Kamino deposit, it transfers the full input amount from the user to the `liquidity_vault`. While currently there's no practical difference between these amounts, this implementation could lead to two impacts:

1. excess tokens being trapped in the `liquidity_vault` that shouldn't have been transferred from the user.

> /// For Kamino banks, the `liquidity_vault` never holds assets, but is instead used as an

`liquidity_vault` is not intended to contain any funds; however, dust accumulation will occur over time.

2. At the moment, there is a check for collateral amount [link](https://github.com/mrgnlabs/marginfi-v2-internal/blob/61dd5fba4ffaf11308feeccbcd6e97f3cdb1dbc/programs/marginfi/src/instructions/kamino/deposit.rs#L77-L82). It uses input amount and allows 1 diff; however, since it must use the actual deposit amount calculated from Klend, this 1 diff may not be sufficient and may result in errors in some cases. I conducted some unit tests but was unable to see the diff greater than 1, even though it is entirely based on Klend invariants.

Location

https://github.com/mrgnlabs/marginfi-v2-internal/blob/61dd5fba4ffaf11308feeccbccd6e97f3cdb1dbc/programs/marginfi/src/instructions/kamino/deposit.rs#L68-L69

https://github.com/mrgnlabs/marginfi-v2-internal/blob/61dd5fba4ffaf11308feeccbccd6e97f3cdb1dbc/programs/marginfi/src/instructions/kamino/deposit.rs#L77-L82

• Klend:

https://github.com/Kamino-Finance/klend/blob/812eb90c6e7da9272494f0660f44897ba5eb81d9/programs/klend/src/handlers/handler_deposit_reserve_liquidity_and_obligation_collateral.rs#L61 https://github.com/Kamino-Finance/klend/blob/812eb90c6e7da9272494f0660f44897ba5eb81d9/programs/klend/src/handlers/handler_deposit_reserve_liquidity_and_obligation_collateral.rs#L91-L94

Mitigation Suggestion

It's better to transfer the specific amount from the user that will be forwarded from liquidity_vault to Klend, matching how Klend handles deposits.

Remediation

This issue has been accepted as `wontfix`, as the accumulation of dust is expected to be insignificant.

ID	ACC-I1
Title	Missing Associated Token Account Validation in `KaminoHarvestReward`
Severity	info
Status	fixed

Description

We found an issue in the `KaminoHarvestReward` function where it fails to validate that `destination_token_account` is an Associated Token Account (ATA). Currently, the function only verifies ownership by the global fee admin, allowing users to potentially provide any token account instead of the proper ATA.

Location

https://github.com/mrgnlabs/marginfi-v2-internal/blob/61dd5fba4ffaf11308feeccbccd6e97f3cdb1dbc/programs/marginfi/src/instructions/kamino/harvest_reward.rs#L42-L47

Relevant Code

```
/// Destination token account must be owned by the global fee admin
    #[account(
        mut,
        constraint = destination_token_account.owner == fee_state.load()?.global_fee_admin @ MarginfiError
::Unauthorized
    )]
    pub destination_token_account: Box<InterfaceAccount<'info, TokenAccount>>,
```

Mitigation Suggestion

Implement proper account derivation validation similar to other functions in the codebase. A reference implementation can be found in the `collect_bank_fees` function:

https://github.com/mrgnlabs/marginfi-v2-internal/blob/61dd5fba4ffaf11308feeccbccd6e97f3cdb1dbc/programs/marginfi/src/instructions/marginfi_group/collect_bank_fees.rs#L30-L37

Remediation

Fixed in commit `a262057371036148 35ff6299c76695e3224e3fb7`.

APPENDIX

Vulnerability Classification

We rate our issues according to the following scale. Informational issues are reported informally to the developers and are not included in this report.

Severity	Description
Critical	Vulnerabilities that can be easily exploited and result in loss of user funds, or directly violate the protocol's integrity. Immediate action is required.
High	Vulnerabilities that can lead to loss of user funds under non-trivial preconditions, loss of fees, or permanent denial of service that requires a program upgrade. These issues require attention and should be resolved in the short term.
Medium	Vulnerabilities that may be more difficult to exploit but could still lead to some compromise of the system's functionality. For example, partial denial of service attacks, or such attacks that do not require a program upgrade to resolve, but may require manual intervention. These issues should be addressed as part of the normal development cycle.
Low	Vulnerabilities that have a minimal impact on the system's operations and can be fixed over time. These issues may include inconsistencies in state, or require such high capital investments that they are not exploitable profitably.
Informational	Findings that do not pose an immediate risk but could affect the system's efficiency, maintainability, or best practices.

Audit Methodology

Accretion is a boutique security auditor specializing in Solana's ecosystem. We employ a customized approach for each client, strategically allocating our resources to maximize code review effectiveness. Our auditors dedicate substantial time to developing a comprehensive understanding of each program under review, examining design decisions, expected and edge-case behaviors, invariants, optimizations, and data structures, while meticulously verifying mathematical correctness—all within the context of the developers' intentions.

Our audit scope extends beyond on-chain components to include associated infrastructure, such as user interfaces and supporting systems. Every audit encompasses both a holistic protocol design review and detailed line-by-line code analysis.

During our assessment, we focus on identifying:
- Solana-specific vulnerabilities
- Access control issues
- Arithmetic errors and precision loss
- Race conditions and MEV opportunities
- Logic errors and edge cases
- Performance optimization opportunities
- Invariant violations
- Account confusion vulnerabilities
- Authority check omissions
- Token22 implementation risks and SPL-related pitfalls
- Deviations from best practices

Our approach transcends conventional vulnerability classifications. We continuously conduct ecosystem-wide security research to identify and mitigate emerging threat vectors, ensuring our audits remain at the forefront of Solana security practices.