SparkFi V12 Audit Report

Smart Contract Security Assessment

Date of Engagement: July 15th, 2024 — July 22nd, 2024\ **Audited by**: Safe Edges

---

Summary

Safe Edges conducted a thorough smart contract audit of SparkFi V12. The assessment involved both automated scanning and manual code review techniques to uncover any vulnerabilities and provide recommendations.

Audit Findings

1. Missing Persistent Update in `increase_user_volume`

Severity: High

Description: In `increase_user_volume`, the logic performs an in-memory update but fails to persist the updated state in storage. The expected behavior is to persist changes after calculating and updating the user's volume.

Impact: The lack of persistence means user trading volume tracking will be inaccurate.

Recommendation: Ensure that the updated `user_volume` is written back to storage.

Status: Unresolved

---

2. Front-Running Vulnerability in Matching Functions

Severity: High

Description: Functions like `match_market_order` and `match_ioc_order` are public and can be front-run. An attacker can predictably exploit these by observing the mempool and preempting orders.

Impact: Results in market manipulation or preferential order execution.

Recommendation: Add mechanisms to protect against front-running, such as commit-reveal schemes or MEV-resistant designs.

Status: Unresolved

---

3. Missing Restriction for IOC Orders

Severity: Medium

Description: The `match_ioc_order` function can be called with limit orders too, which breaks the intent of IOC behavior.

Impact: Incorrect order execution logic.

Recommendation: Add type-checking to ensure only IOC orders are passed.

Status: Unresolved

---

4. Lack of Vector Length Restriction

Severity: Medium

Description: Several functions take in `Vec<T>` inputs but do not restrict their maximum length.

Impact: Potential denial-of-service from large inputs or excessive gas use.

Recommendation: Set and enforce maximum length for vectors in function parameters.

Status: Unresolved

---

5. Missing Documentation

Severity: Low

Description: Several functions lack clear documentation or NatSpec comments.

Impact: Poor developer experience and risk of misunderstanding the intended behavior.

Recommendation: Add descriptive comments and usage examples.

Status: Unresolved

---

6. `log_order_change_info` Emits No Logs

Severity: Low

Description: The `log_order_change_info` function is defined and called but emits no log.

Impact: Pointless function execution and wasted gas.

Recommendation: Either remove the function or ensure it emits meaningful logs.

Status: Unresolved

---

7. Dead Code in `mul_div_rounding_up`

Severity: Low

Description: The `mul_div_rounding_up` function is never used.

Impact: Increased contract size and audit overhead.

Recommendation: Remove unused code.

Status: Unresolved

---

8. Ineffective `require` in `order_id`

Severity: Low

Description: The `require` condition `require(token_id_a != token_id_b)` in `order_id` may be trivial since validation is already assumed elsewhere.

Impact: Code redundancy.

Recommendation: Evaluate necessity and remove if truly redundant.

Status: Unresolved

Conclusion

The SparkFi V12 contracts are well-structured but exhibit key security oversights involving storage persistence, front-running vectors, missing validations, and documentation gaps. Prompt implementation of the above recommendations is essential to enhance the protocol's security posture.

Safe Edges remains available for a follow-up audit to verify remediations.

Report Prepared By:\ **Safe Edges Security Team**\ https://safeedges.in