Streamflow

Audit Report

MOVEBIT

contact@movebit.xyz

https://twitter.com/movebit_

Fri Mar 08 2024

Streamflow Audit Report

1 Executive Summary

1.1 Project Information

Description	A crypto asset streaming protocol.
Type	DeFi
Auditors	MoveBit
Timeline	Wed Mar 06 2024 - Fri Mar 08 2024
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/streamflow-finance/sui-streamflow-module
Commits	37be5819901fa0c2c0b8d3976b3666393b57878a 5de97f90233d1857539c1a8feee59128c3278f15

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
TUT	sources/test_utils.move	2bd0a3cfdab296f0cb7dba28d7f13e2d04cecd1a
STR	sources/strmt.move	87e2f9e0065c5bec7e13006675a475bc8b774478
UTI	sources/utils.move	0863762fa4dae6b77a2430f0ed9baefb54e90e74
FMA	sources/fee_manager.move	bbb5730f0e6b9352c1c7e55e2e0294834596e01d
ADM	sources/admin.move	6ed3bec51b4f9b28d1b640cb4bd01e7a66fc5196
PRO	sources/protocol.move	1122d6808a61ac3b7218c11727d9a6559eaa6ca0

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	2	2	0
Informational	1	1	0
Minor	1	1	0
Medium	0	0	0
Major	0	0	0
Critical	0	0	0

1.4 MoveBit Audit Breakdown

MoveBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow by bit operations
- Number of rounding errors
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting
- Unchecked CALL Return Values
- The flow of capability
- Witness Type

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Formal Verification

Perform formal verification for key functions with the Move Prover.

(4) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by Streamflow to identify any potential issues and vulnerabilities in the source code of the Streamflow smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 2 issues of varying severity, listed below.

ID	Title	Severity	Status
ADM-1	Inappropriate Event Parameters	Minor	Fixed
ADM-2	Incorrect Comment	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the Streamflow Smart Contract :

Admin

- Admin can update the `streamflow_fee` and `tx_fee` through `change_tx_fee` and `change_streamflow_fee` .
- Admin can update the `treasury` address through `change_treasury` .
- Admin can add a `FeeValue` to `Fee_Table` through `fees_write` .

User

- User can create a `Contract` through `create` .
- User can update the information of `Contract` through `update` .
- User can get the `Contract` coin through `withdraw` .
- User can cancel the `Contract` and withdraw the left coin through `cancel` .
- User can extend the Contract time through `topup` .
- User can pause and unpause the `Contract` through `pause` and `unpause` .

4 Findings

ADM-1 Inappropriate Event Parameters

Severity: Minor

Status: Fixed

Code Location:

sources/admin.move#100

Descriptions:

The event emitted by the `change_fee_manager` function when it is first called is `event::emit(FeeManagerChanged { new_address: new_fee_manager, old_address: tx_context::sender(ctx) });` , which may be confused with the events emitted by subsequent calls that modify existing values.

Suggestion:

It is recommended to change the value of `old_address` to the zero address.

Resolution:

The client has changed the value of `old_address` to a zero address.

ADM-2 Incorrect Comment

Severity: Informational

Status: Fixed

Code Location:

sources/admin.move#10

Descriptions:

The value of the `MAX_FEE` parameter is set to 100%, but the comment says 0.5%.

Suggestion:

It is recommended to modify the comments of the `MAX_FEE` parameter correctly.

Resolution:

The client has already addressed this issue.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

MOVEBIT VERIFIED

MOVEBIT