Bucket Interest Rate

Audit Report

MOVEBIT

contact@movebit.xyz

https://twitter.com/movebit_

Mon Feb 05 2024

Bucket Interest Rate Audit Report

1 Executive Summary

1.1 Project Information

Description	This update adds the new feature of interest rate to the bucket protocol.
Type	DeFi
Auditors	MoveBit
Timeline	Tue Jan 30 2024 - Mon Feb 05 2024
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/Bucket-Protocol/v1-core/
Commits	29ef62b4f8c600293e27cb5e33352b128d77f375 dccc984c7b20f1c42b6e8ba0a2ebe466eb128e8b

1.2 Files in Scope

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

ID	File	SHA-1 Hash
VLO	framework/sources/vesting_lock.move	a6fccadf49e3d13e8d2dd8bd02cda7a4d855d67e
LTA	framework/sources/linked_table.move	0169cd8710d12220fdaa53acc4c95aa177c7d85e
REV	protocol/sources/events/reservoir_events.move	d9eef37ea26c63a8b02856296297b80c8f2037e0
BEV1	protocol/sources/events/bucket_events.move	71d3fb2e8dbf2d38126e448b7c1eb0062db0cfa0
TEV	protocol/sources/events/tank_events.move	8dce2891977ce26b7a9134246401f63b264bb27f
WEV	protocol/sources/events/well_events.move	d54fc22da7bdfb32aaa91dc8e7245dcfc3e934e2
WEL	protocol/sources/well.move	d7f1acfee9e4c2431ca54e21c3e17ffe0b869c52
TAN	protocol/sources/tank.move	f317209a74425cdf65043781b47c0a4fb81eac2d
BKT	protocol/sources/bkt.move	f5b1fff63a4510c7e1c1870ac603ee2fdcb9f00d
MAT	framework/sources/math.move	57f7af4d64a6bcd830375ad9dd2dce7438c4a001
RES	protocol/sources/reservoir.move	3356df01d0ded178b9f70b3a96fa331b043bb87f

BEV	protocol/sources/events/buck_events.move	47ff16b09d22b422156b28bfcbf13dc08ff6c9e4
CON	protocol/sources/config/constants.move	fe87760ce8162e79c0dfe4855c7a0cfc2ac909d9
BOT	protocol/sources/bottle.move	b782167863cad15e602d92e64ff6f4f841ce7e50
ITA	protocol/sources/lib/interest_table.move	dcb98112c74048b7f6f000bec727c712f2bdb27a
BUC	protocol/sources/buck.move	dce51b8f9f85d96a2da76da1cfd43d81235d313c
BUC1	protocol/sources/bucket.move	141c69784578cbff443bcadddfdfd2f2134d13c4

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	4	4	0
Informational	1	1	0
Minor	2	2	0
Medium	1	1	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 Bucket to identify any potential issues and vulnerabilities in the source code of the bucket_protocol 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 4 issues of varying severity, listed below.

ID	Title	Severity	Status
BOT-1	`cr_greater_with_interest` and `cr_less_or_equal_with_interest` Checks Wrong Bottle	Medium	Fixed
BUC-1	Lack of Events Emit	Minor	Fixed
BUC-2	`set_interest_rate` Should Check The Range Of `new_interest_rate`	Minor	Fixed
ITA-1	Do Not Use Hard-coded Error For `calculate_interest_index`	Informational	Fixed

3 Participant Process

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

Admin

- `admin` can set interest rate through `set_interest_rate`.

User

- `user` can get the interest table info through `get_interest_table_info`.
- `user` can get the bottle interest index `get_bottle_interest_index`.
- `user` can get the interest rate through `get_interest_rate`.

4 Findings

BOT-1 `cr_greater_with_interest` and `cr_less_or_equal_with_interest` Checks Wrong Bottle

Severity: Medium

Discovery Methods: Manual

Status: Fixed

Code Location:

protocol/sources/bottle.move#607-622;

protocol/sources/bottle.move#631-646

Descriptions:

In the `cr_greater_with_interest` function, it is used to compare two `bottle`'s collateral ratios.

However, the second `if` condition is not correct. It should be checking the condition of `bottle_cmp` instead of `bottle` to get its info.

Checking the wrong `bottle` may cause the function to panic.

The same issue applies to the `cr_less_or_equal_with_interest` function.

Suggestion:

It is suggested to check the `bottle_cmp` in the second if condition.

Resolution:

It is fixed by the client to change the `bottle` to `bottle_cmp`.

BUC-1 Lack of Events Emit

Severity: Minor

Discovery Methods:

Status: Fixed

Code Location:

protocol/sources/buck.move#942-1022;

protocol/sources/lib/interest_table.move#29-63

Descriptions:

The smart contract lacks appropriate events for monitoring sensitive operations, which could make it difficult to track sensitive actions or detect potential issues.

Suggestion:

It is recommended to emit events for those sensitive functions.

Resolution:

It is fixed by the client to add the events.

BUC-2 `set_interest_rate` Should Check The Range Of `new_interest_rate`

Severity: Minor

Discovery Methods: Manual

Status: Fixed

Code Location:

protocol/sources/buck.move#980-995

Descriptions:

In `set_interest_rate` function, new interest rate is set using `new_interest_rate_apr,constants::interest_precision(), 10000 * constants::ms_in_year()`. However, the last two parameters are constants and require no checks.

`new_interest_rate_apr` is a parameter that can be set arbitrarily.

If `new_interest_rate_apr` is set over 100%, then it's entirely users' loss. And there are no checks in either `set_interest_rate` in `buck.move` or `interest_table.move` Nor would `mul_factor_u256` panic.

Suggestion:

It is suggested to limit the `new_interest_rate_apr` in a reasonable range.

Resolution:

It is fixed by the client to limit the interest rate to under 10000 (which is 100%).

ITA-1 Do Not Use Hard-coded Error For `calculate_interest_index`

Severity: Informational

Discovery Methods: Manual

Status: Fixed

Code Location:

protocol/sources/lib/interest_table.move#97

Descriptions:

In the `calculate_interest_index` it's a good practice to check the timestamp, however, it's not recommended to use the raw 0 for error info.

```
assert!(clock::timestamp_ms(clock) >= last_active_index_update_cache, 0);
```

As if the assertion fails, the cause is not clear.

Suggestion:

It is suggested to add a `const` error and use it in the assertion.

Resolution:

It is fixed by the client by changing the hard-coded value into a const error.

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

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.