MOVEBIT
Securing the Move Ecosystem

TurboStar Smart Contract **Audit Report**

https://twitter.com/movebit_

contact@movebit.xyz

TurboStar Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	A simple launchpad project on Sui.
Type	Launchpad
Auditors	MoveBit
Timeline	May 6, 2023 – May 15, 2023
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/turbos-finance/turbos-presale
Commits	297c0ff8e9b1dd52e970fe282fd7cb902696f537 7cd3aaa328b86769935490d90fad464473899d80 52f292ae5331dd9f795bffbe66441950aa58a977

1.2 Files in Scope

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

ID	Files	SHA-1 Hash
IDO	sources/ido.move	ea860df6cbb24b393fb57c9132ced092a9606b4c
CLAIM	sources/claim.move	9a495588b117351320047ee243cd86837a150fde

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	9	8	1
Informational			
Minor	6	6	
Medium	3	2	1
Major			
Critical			

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 **Turbos** to identify any potential issues and vulnerabilities in the source code of the **TurboStar** 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 have identified **9** issues of varying severity, listed below.

ID	Title	Severity	Status
IDO-1	Parameter Validation is Missing When Creating a `PreSale`	Minor	Fixed
IDO-2	Redundant Field Attributes in a Struct	Minor	Fixed
IDO-3	Function Optimization	Minor	Fixed
IDO-4	Sensitive Operation Lacks `Event`	Minor	Fixed
IDO-5	Incorrect Usage of Assert Error Code	Minor	Fixed
IDO-6	Missing Function to Remove Users from Whitelist	Medium	Fixed
IDO-7	The Function's Functionality Does Not Match its Naming	Minor	Fixed
IDO-8	Logical Loophole	Medium	Acknowledged
CLAIM-1	Incorrect Data in Event	Medium	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the `turbos-presale` Smart Contract：

Creator

- Creator and users can create `PreSale` through `create_presale()`.
- Creator can transfer funds from the `PreSale` through `transfer_funds()` and `transfer_funds_to_self()`.
- Creator can limit whether only whitelisted users can participate in the `PreSale` through `set_pub_or_whitelisted_only()`.
- Creator can add whitelist addresses through `add_white_list()`.
- Creator can remove whitelist addresses through `delete_white_list()`.
- Creator can update the end time through `change_end_time()`.
- Creator can update the start time through `change_start_time()`.
- Creator can create `Claim` through `create_claim()`.
- Creator can add the address and token amount of the claiming user through `add_wait_claim_list()`.
- Creator can switch whether it is currently an emergency through `emergency_switch()`.
- Creator can withdraw the token in an emergency through `emergency_withdraw()`.
- Creator can deposit tokens in an emergency through `emergency_depost()`.

User

- User can deposit funds through `fund()`.
- User can check whether the address is in the whitelist through `check_whitelisted()`.
- User can check whether the address is in `PreSale` members through `check_funded()`.
- User can update the amount limit in `PreSale` through `change_fund_amount()`.
- User can update the `raise` in `PreSale` through `change_raise()`.
- User can claim tokens through `claim()`.

4 Findings

IDO-1 Parameter Validation is Missing When Creating a `PreSale`

Severity: Minor

Status: Fixed

Code Location: ido.move#L53, L172 ,L181.

Descriptions: In the function `create_presale()`, there is a lack of validation for the parameters `start_time` and `end_time`. The `start_time` should be greater than or equal to the current time and less than the `end_time`. The functions `increment_endtime()` and `increment_starttime()` also have the same issue.

Suggestion: Add assertion statements to validate the parameters.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-2 Redundant Field Attributes in a Struct

Severity: Minor

Status: Fixed

Code Location: ido.move #L35.

Descriptions: The `status` field in the `PreSale` struct is not used.

Suggestion: Remove the `status` field.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-3 Function Optimization

Severity: Minor

Status: Fixed

Code Location: ido.move #L134.

Descriptions: The functions `transfer_funds_to_self()` and `transfer_funds()` have almost identical logic, with just the recipient address being different. To simplify the code, we can directly call the `transfer_funds()` from within the `transfer_funds_to_self()` and pass in the account owner's address.

Suggestion: Directly call the `transfer_funds()` from within the `transfer_funds_to_self()` and pass in their own account address.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-4 Sensitive Operation Lacks `Event`

Severity: Minor

Status: Fixed

Descriptions: Some sensitive operations lack `Event`.

Suggestion: Add `Event` to sensitive operations.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-5 Incorrect Usage of Assert Error Code

Severity: Minor

Status: Fixed

Code Location: ido.move #L103.

Descriptions: The assert error code is used incorrectly. `USER_MAX_CAP_REACHED` was used instead of `USER_MIN_CAP_REACHED`.

Suggestion: Replace `USER_MAX_CAP_REACHED` with `USER_MIN_CAP_REACHED`.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-6 Missing Function to Remove Users from Whitelist

Severity: Medium

Status: Fixed

Descriptions: Currently, there is a function to add users to the whitelist, but there is a lack of a function to remove users from the whitelist. If an admin has added a wrong user to the whitelist, they cannot remove them afterward.

Suggestion: Add the function to remove users from the whitelist.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-7 The Function's Functionality Does Not Match its Naming

Severity: Minor

Status: Fixed

Code Location: ido.move #L172, L181.

Descriptions: The function's functionality does not match its naming. The actual function updates the time, but the function name implies that it adds time.

Suggestion: Change the function name to `update_endtime` and `update_starttime` accordingly.

Resolution: The client has followed our suggestion and fixed the issue.

IDO-8 Logical Loophole

Severity: Medium

Status: Acknowledged

Code Location: ido.move #L134, L166.

Descriptions: If `sale.balance` has already reached the `raise` goal, the IDO should have ended, but by using `transfer_funds()` to transfer `sale.balance` and reducing it below the `raise` goal, users can then deposit tokens through the function `fund()`, which is like giving more people the opportunity to participate in the IDO.

Suggestion: Recording the number of tokens deposited and not reducing it with token transfers can help ensure that once the deposited amount reaches `sale.raise`, the IDO will end.

Resolution: The client doesn't think it's a problem and doesn't fix it.

CLAIM-1 Incorrect Data in Event

Severity: Medium

Status: Fixed

Code Location: claim.move #L83.

Descriptions: There is a calculation error in the `claim()` function, where the quantity in the event is always 0.

Suggestion: Please check again and recalculate the `amount`.

Resolution: The client has followed our suggestion and fixed the 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

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.

MOVEBIT

Securing the Move Ecosystem

https://twitter.com/movebit_

contact@movebit.xyz