Token Deployer Smart Contract

Audit Report

MOVEBIT

contact@movebit.xyz

https://twitter.com/movebit_

Tue Jan 09 2024

Token Deployer Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	A coin deployer
Type	Token
Auditors	MoveBit
Timeline	Mon Jan 08 2024 - Mon Jan 08 2024
Languages	Move
Platform	Aptos
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/BAPTSWAP/TokenDeployer-v1
Commits	1477b9c46d1027be15c02cfe41c3e78ae3701419 4e6f6a08d6739c6e17fa89332bfebaa521dcc127

1.2 Files in Scope

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

ID	File	SHA-1 Hash
INI	build/CoinDeployer/sources/init.move	5456689bda35d8eb4f5e4ed945f23097e2cf96bb
FEE	build/CoinDeployer/sources/fee.move	484324145d00e469b2d5e6416dec88aa949dc757
MOV	Move.toml	40eac91c698d603b8f6928f06bfcfcd207c29710
DEP1	sources/deployer.move	2272623a296682a4c8e7440fa60680a7a8fb1df6

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 Baptswap to identify any potential issues and vulnerabilities in the source code of the Token Deployer 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
DEP-1	Lack of Events Emit	Minor	Fixed
DEP-2	Incorrect Use of Error Code	Informational	Fixed

3 Participant Process

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

Admin

- The Admin can initialize the deployer contract through `init()` .
- The Admin can update the fee of the contract through `update_fee()` .
- The Admin can update the fee collector address through `update_owner()` .

User

- The User can generate a new coin and mint the total supply to the deployer through `generate_coin<CoinType>()` .

4 Findings

DEP-1 Lack of Events Emit

Severity: Minor

Status: Fixed

Code Location:

sources/deployer.move#36,41,53;

build/CoinDeployer/sources/fee.move#14,34,40

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:

The client followed our suggestion and fixed this issue.

DEP-2 Incorrect Use of Error Code

Severity: Informational

Status: Fixed

Code Location:

sources/deployer.move#75

Descriptions:

The error code used for the assert of `config` in the `generate_coin` function should be consistent with the preceding code.

Suggestion:

It is recommended to modify the error code `ERROR_ERROR_INSUFFICIENT_APT_BALANCE` as `ERROR_INVALID_BAPT_ACCOUNT`.

Resolution:

The client followed our suggestion and fixed 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