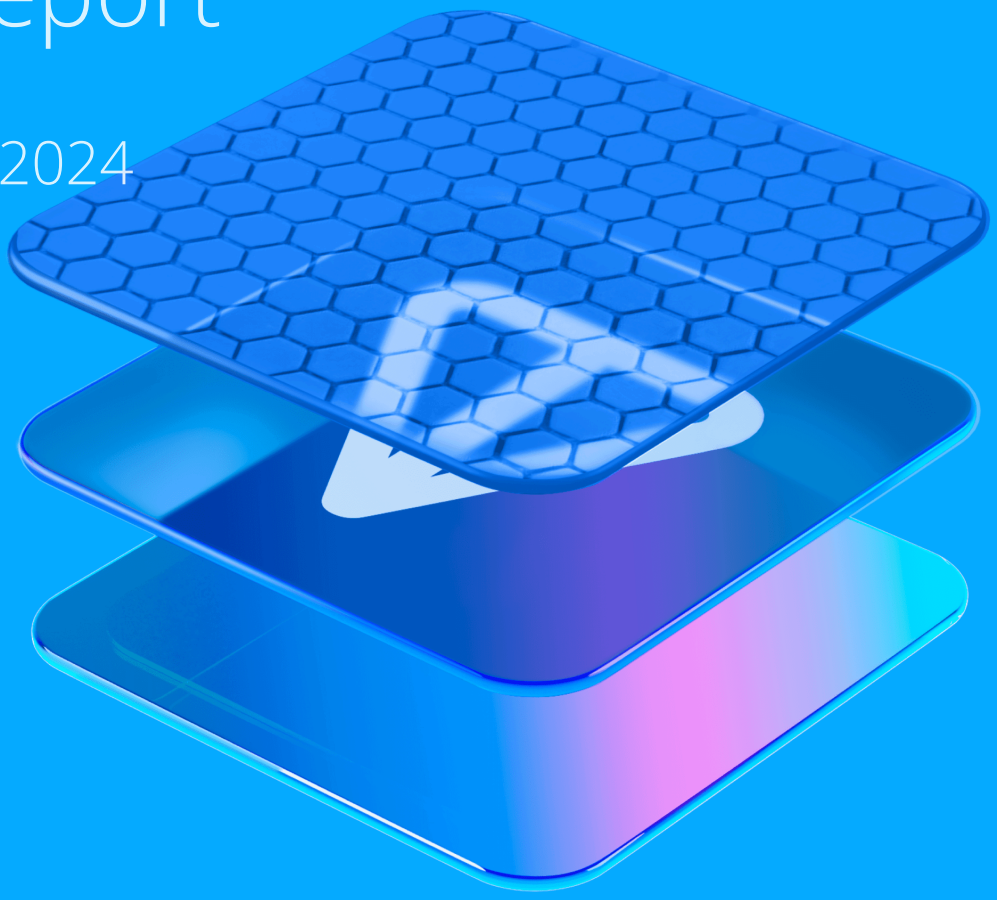
One Click Sender Audit Report

Tue Aug 20 2024



contact@bitslab.xyz https://twitter.com/tonbit_

TonBit

One Click Sender Audit Report

1 Executive Summary

1.1 Project Information

Description	A protocol for batching the sending of Jetton and TON
Type	DeFi
Auditors	TonBit
Timeline	Tue Jul 23 2024 - Tue Aug 20 2024
Languages	FunC
Platform	Ton
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/ModoriLabs/ton-batch-sender.git
Commits	fd2b251374896228fdfdede0aa6d11f877bef6a6 d28684969f10b443c2eedf9bcba35f81eacc907f 9083db940e6c4633d48294e9db042787e265017b 8c8b8a380363f7506ea886926b2e7a7abfb8ee04

1.2 Files in Scope

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

ID	File	SHA-1 Hash
GAS	contracts/gas.fc	3b18710a537435c4c353f25ed0cd8e39459129e7
ERR	contracts/error.fc	d090d3f99435c22fd910ea0cc627e2dd84195cd1
STD	contracts/imports/stdlib.fc	2f104cd568a4cebb1c4112ecf8979800f0672575
MES	contracts/message.fc	77bfd9e186e4591a80273b8117b9adf685059cb0
OP	contracts/op.fc	2542e1adfd34268b49ea9cb76e768066f4af9715
BSE	contracts/batch_sender.fc	4c6afd80b6b27504dc1a54c1e9a38d559c239121
STO	contracts/storage.fc	a676fb49096ad4ce8a2f5b10083b1739e1ec3a62

1.3 Issue Statistic

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

1.4 TonBit Audit Breakdown

TonBit 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

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** 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) 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 ModoriLabs to identify any potential issues and vulnerabilities in the source code of the One Click Sender 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
BSE-1	Excess Fee Issue in Message Handling	Major	Fixed
BSE-2	Missing Parameter Validation	Major	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the One Click Sender Smart Contract :

Owner

- The owner can set one-time fee through `set_one_time_fee()`
- The owner can set per-user fee through `set_per_user_fee()`
- The owner can set fee receiver address through `set_fee_receiver_address()`
- The owner can set referral discount through `set_referral_discount()`
- The owner can set referral fee through `set_referral_fee()`
- The owner can withdraw native currency through `withdraw_native()`
- The owner can withdraw through `withdraw()`

User

- The user can send transfer notification through `transfer_notification()`
- The user can send TON through `send_ton()`

4 Findings

BSE-1 Excess Fee Issue in Message Handling

Severity: Major

Status: Fixed

Code Location:

contracts/batch_sender.fc#254

Descriptions:

When the contract handles `op::transfer_notification` messages, it sends the `modified_cost` to the `fee_receiver_address`. However, when processing `op::send_ton` messages, it directly sends the `cost` to the `fee_receiver_address`. This is incorrect as it results in sending excess fees.

```
if (cost > 0) {
    int modified_cost = cost;
    if (has_referral) {
        int referral_fee = cost * storage::referral_fee / 100;
        slice referral = in_msg_body~load_msg_addr();
        modified_cost = cost - referral_fee;
        send_empty_message(referral_fee, referral, IGNORE_ERRORS);
    }

    send_empty_message(cost, storage::fee_receiver_address, IGNORE_ERRORS);
}
```

Suggestion:

It is recommended to change the code to send `modified_cost` instead of `cost` when handling `op::send_ton` messages, to prevent sending excess fees.

Resolution:

This issue has been fixed. The client has changed the code to send `modified_cost` instead of `cost` when handling `op::send_ton` messages.

BSE-2 Missing Parameter Validation

Severity: Major

Status: Fixed

Code Location:

contracts/batch_sender.fc#131-260

Descriptions:

The current contract does not validate parameters like `has_referral` and `referral` when handling `op::send_ton` and `op::transfer_notification` operations. This allows users to manipulate these parameters arbitrarily, which can disrupt the contract's logic. For example, a user could set the `referral` to their own address to evade a portion of the fees.

```
if(has_referral) {
  discount = storage::referral_discount;
}

var cost = 0;

if(fee_type) {
  cost = storage::per_user_fee * len;
} else {
  cost = storage::one_time_fee;
}

return cost - (cost * discount / 100);
```

Suggestion:

It is recommended to add additional logic to validate these parameters.

Resolution:

This issue has been fixed. The client has added additional logic to validate these parameters.

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.