

FIVA Audit Report

Tue Mar 04 2025



contact@bitslab.xyz



https://twitter.com/tonbit_



FIVA Audit Report

1 Executive Summary

1.1 Project Information

Description	
Type	DeFi
Auditors	TonBit
Timeline	Thu Jan 23 2025 - Mon Feb 24 2025
Languages	FunC
Platform	Ton
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/Fiva-protocol/contracts_v2
Commits	94be481a84148e40ef7d6febf07240de75cdd5fc0454c9ede10aa7de4f4c911776f49ccbb20f8a70

1.2 Files in Scope

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

ID	File	SHA-1 Hash
SEN	contracts/YT/send.fc	e653cbd4f7b89edfb1621c3abf37e8b25eb6d667
BUR	contracts/YT/burn.fc	465b84ab1fa71250bab19b28ce965daa9a6aa4ee
INT	contracts/YT/interest.fc	8220cc4de2dd067e9c2245d18425a722ce6b8e4f
MAI	contracts/YT/maintance.fc	bfe655f00787a7fe9e1777a600f5fe8d0deb34cd
STO	contracts/YT/storage.fc	45317e9745c2933dcf02dedc470ccc0b3ea5755
GET	contracts/YT/getters.fc	9163add6ee23618093aaaca70f57cb3a95d72e21
DEP	contracts/YT/redeem/deposit.fc	3134fd259b07ed653bb387d368135585a97c976d
UTI	contracts/YT/redeem/utls.fc	a5f7ae262add4369e3b1fd06c1f83d31cae6d7e1
RED	contracts/YT/redeem.fc	ec77245cd7408ba062f967fff93a9cef6b455b4b
WAL	contracts/YT/wallet.fc	4ec8fa73dd1dbe0f7a7d4dd9911de90a1c41c5ed
MIN	contracts/YT/minter.fc	5f84a694aaaa3cbb852a7e792c3a33e0efd28ede

FEE	contracts/YT/fees.fc	542db3bef55b470e8de39f6a84fd6e66f4a1ffc8
MIN1	contracts/YT/mint.fc	975f3548281fd28862541fbb4fe546d44c122a37
WAL1	contracts/PT/wallet.fc	6fc537931cae0662e7ab6cb4d210e4ccdb2d6810
MIN2	contracts/PT/minter.fc	da7832624470ef73e98336bcbe4cb2249c32a67
MAI1	contracts/SY/maintance.fc	eb607f6f8735dc0fd83c75b0b513b4343bd90f49
STO1	contracts/SY/storage.fc	a2677183973a2ac0c98703b71a38d04f3846ce08
GET1	contracts/SY/getters.fc	52662a8bffafffd92a985882edc0f7aa90924f66
WAL2	contracts/SY/wallet.fc	778cadea394abb99f77a3deb702cd37169e75d0e
MIN3	contracts/SY/minter.fc	b3408bf63d028a409f76c878a7b7d9d3bdf9efbf
FEE1	contracts/SY/fees.fc	4d5a63eb0428d1ca2f94bba705cc0c52a8189985
WAL3	contracts/stake/wallet.fc	7dc0d91cb716af753763e4e5ff18f7cc1b517bef
MIN4	contracts/stake/minter.fc	22465d9272deae5d82445e123a9c94903dfb03b7
LIQ	contracts/AMM/liquidity.fc	9189c1a1e1bf43022b1329c3e134763031224d9f

MAI2	contracts/AMM/maintance.fc	9ce3dc30972a8c82942a0e36e62130ed6a374e8b
STO2	contracts/AMM/storage.fc	7bd952c2c2e82cd103ba48d437e177501f868036
GET2	contracts/AMM/getters.fc	4f5a259f67d9577892588554636023b3b49d7b0c
PAR	contracts/AMM/params.fc	2f1060c75c99062160b4f5fc11d0d7ca6dc1cc23
FEE2	contracts/AMM/fee.fc	8f80fe513c18280f799af4f4057c5c3b3fc0bca7
POO	contracts/AMM/pool.fc	33a83e84f84ceefe46e08cf83269f5c344079528
ECO	contracts/AMM/error-codes.fc	a2456266f7ec69fb387ebefac57920ba68c7d10a
OCO	contracts/AMM/op-codes.fc	841715a92f6ce0bcbf1292f02db61306beb05eb4
DEP1	contracts/AMM/lp_deposit/deposit.fc	dfaf7896b18d25794e6555c6e6e95998bf815346
UTI1	contracts/AMM/lp_deposit/utills.fc	715a3a3f8c8a67483f52b6ab2e57574c5ced43fa
LWA	contracts/AMM/lp_wallet.fc	db6ae4997bf367bd7ca09ccfedb64bd6c24f451a
UTI2	contracts/AMM/utills.fc	9b62bf69da9bc3e9ca44c9931aff035e610f6426
YHE	contracts/AMM/markets/yt_helpers.fc	279e02dc426c9e0da547a3c14b8855c2e52b505d

SWA	contracts/AMM/markets/swaps.fc	f24e8117226f768be173be8d67a6cb1885ecca83
MAT	contracts/AMM/markets/const_product/math.fc	433d2dd5be66ce6ad9e73c2dcfada39192a7ce4e
MAT1	contracts/AMM/markets/curve_stable/math.fc	dbbb21905e13e1bba4b593900be0ed5aa8d3bce7
MAT2	contracts/AMM/markets/cube_stable/math.fc	710c75d9d25cdb6a5d6bf3fd9886c92aa9a0f4b3
COL	contracts/nft-access/collection.fc	b790fdbad307293590157da703b90aacf4492033
ITE	contracts/nft-access/item.fc	e758c94bf6a5e4a5a21878492cfe8022ebea52d9
PRO	contracts/prophet/prophet.fc	45f1e6742809008346fe5d02c616411f0e07f641
STO3	contracts/prophet/storage.fc	1ad6656909de91847ae6a01947aa17634ea6782e
MV2	contracts/unlimited/minterV2.fc	faea75a6e96cc9516bec42cd518f48b442d41628
WAL4	contracts/unlimited/wallet.fc	4d5023e160bbbdaafe147df430e53574d06931c3
MIN5	contracts/unlimited/minter.fc	2cdaacce0e036e4b7a2866ec58aa613e88aa6952
MAT3	contracts/common/math.fc	0ff544c21195f146a3d39d1d88aec33478df9615
PAR1	contracts/common/params.fc	a0b576caafcba47d25810a30e960de3cf87e2b2c

ECO1	contracts/common/error-codes.fc	60c0d0f1e2663c98681659c117cbe fc18098e9e9
STD	contracts/common/stdlib.fc	7b81effc0c96d91ba0d882569ffc3 825891a6e8
OCO1	contracts/common/op-codes.fc	a73304e1a1d527b505b1bb5e0974 80c5792b6209
MES	contracts/common/messages.fc	19c0913dc7c4449b29bf5956518f7 3fc34b0aa90
FEE3	contracts/common/fees.fc	51bbf60dd148f9b199a90dea3eed 814a3d471cf8
JUT	contracts/common/jetton-utils.fc	2bca6d9772684f3e9ccf9c7436341 53337b4c76d

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	4	4	0
Informational	1	1	0
Minor	1	1	0
Medium	2	2	0
Major	0	0	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 [FIVA](#) to identify any potential issues and vulnerabilities in the source code of the [FIVA](#) 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
DEP-1	Should Reset Balance To Zero	Informational	Fixed
MIN-1	Missing <code>IGNORE_ERRORS</code> flag	Minor	Fixed
POO-1	Centralization Risk	Medium	Fixed
SWA-1	Potentially incorrect use of <code>MIN_SWAP_AMOUNT</code> in one of the checks	Medium	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the [FIVA](#) Smart Contract :

Users

- `wrap` : Users wrap their tokens to `SY` .
- `wrap_and_swap_sy_for_pt` : Users wrap their tokens to `SY` ,then swap `SY` for `PT` .
- `unwrap` : Users unwrap their `SY` to specific token.
- `add_liquidity` : Users add `SY` or `PT` to pool to provide liquidity ,and receive some lp tokens.
- `swap_sy_for_pt` : Users swap `SY` for `PT` .
- `swap_pt_for_sy` : Users swap `PT` for `SY` .
- `swap_sy_for_yt` : Users swap `SY` for `YT` .
- `swap_pt_for_yt` : Users swap `PT` for `YT` .This function has not finished yet.
- `swap_yt_for_sy` : Users swap `YT` for `SY` .
- `swap_yt_for_pt` : Users swap `YT` for `PT` .

Pool Admin

- `change_owner` : Change pool's owner.
- `Update Fees` : Update fees such as `lp_fee` , `ref_fee` , `protocol_fee` .
- `update_pool_index` : Update pool index.
- `upgrade_code` : Update the pool contract.
- `upgrade_storage` : Update the pool's all storage.
- Some other admin functions.

Jetton List

- `PT` : Principle Token.
- `YT` : Yield Token.Represents the future yield generated by the asset.
- `SY` : Staked Yield Token.
- `unlimited`

- `stake`
- `lp_stake` : Minted when users add liquidity to pool.

NFT

- `nft-access` : With the function of Merkle Tree Proof

Oracle

- `prophet` : This contract is in development stage.

4 Findings

DEP-1 Should Reset Balance To Zero

Severity: Informational

Status: Fixed

Code Location:

contracts/YT/redeem/deposit.fc#90

Descriptions:

In the file `contracts/YT/redeem/deposit.fc`, after redeeming a user's PT + YT, the balance should be reset to 0. However, this step is missing here. In `send_finalize_redeem()`, the message is sent with `SELFDESTRUCT_ON_EMPTY` mode, so contract should be destroyed there. Though, we should reset balance to 0 just as additional safety measure.

Suggestion:

reset balance to 0 after sending a message.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

MIN-1 Missing IGNORE_ERRORS flag

Severity: Minor

Status: Fixed

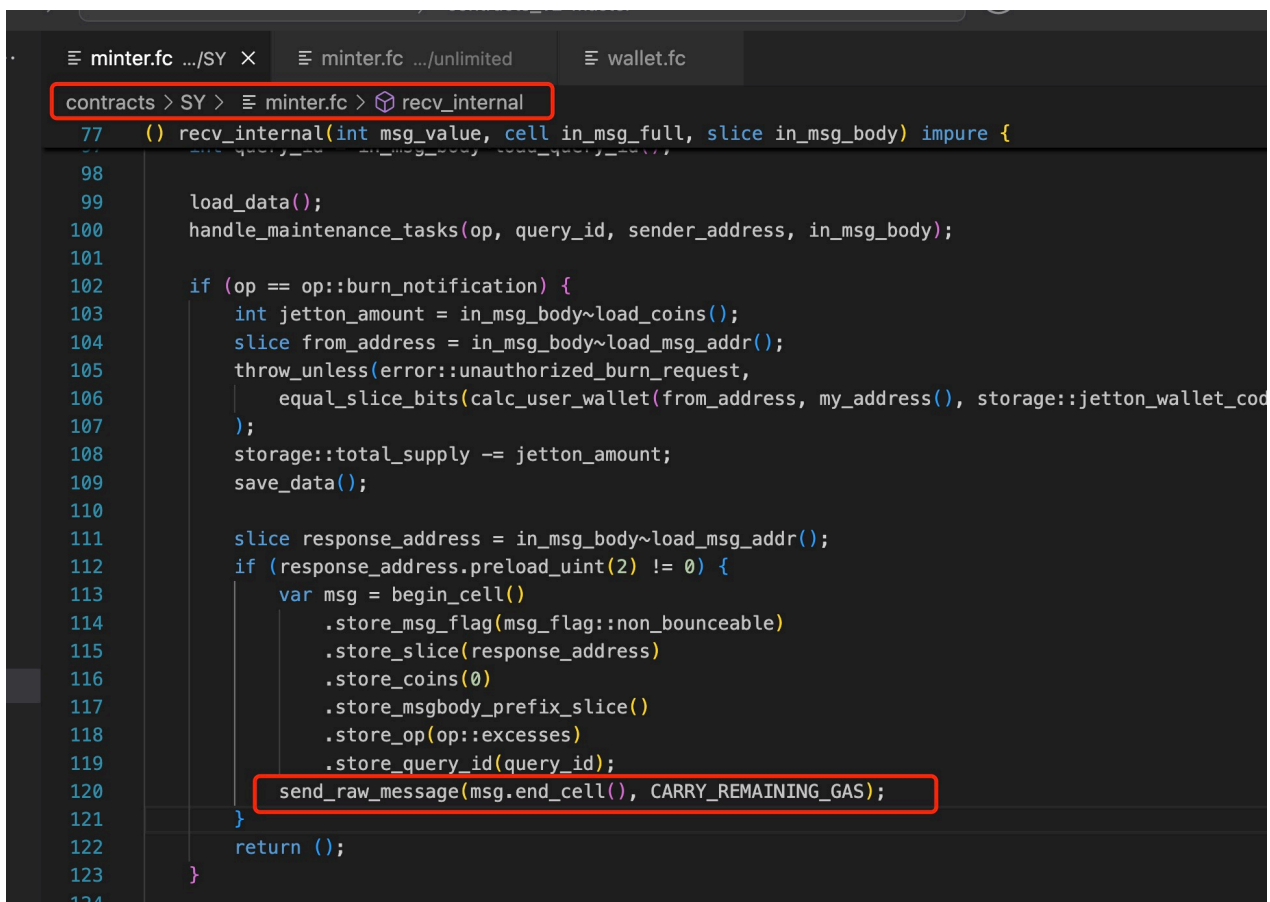
Code Location:

contracts/SY/minter.fc#120

Descriptions:

According to the official Jetton implementation recommendation, when the master receives the `op::burn_notification` message, it needs to send the `op::excesses` message. When sending the message, errors should be ignored to prevent unexpected errors from causing a rollback, which could lead to an inaccurate `storage::total_supply`.

The following is the incorrect part.:



```
contracts > SY > minter.fc > recv_internal
77  () recv_internal(int msg_value, cell in_msg_full, slice in_msg_body) impure {
98
99      load_data();
100     handle_maintenance_tasks(op, query_id, sender_address, in_msg_body);
101
102     if (op == op::burn_notification) {
103         int jetton_amount = in_msg_body~load_coins();
104         slice from_address = in_msg_body~load_msg_addr();
105         throw_unless(error::unauthorized_burn_request,
106                     equal_slice_bits(calc_user_wallet(from_address, my_address(), storage::jetton_wallet_cod
107         ));
108         storage::total_supply -= jetton_amount;
109         save_data();
110
111         slice response_address = in_msg_body~load_msg_addr();
112         if (response_address.preload_uint(2) != 0) {
113             var msg = begin_cell()
114                 .store_msg_flag(msg_flag::non_bounceable)
115                 .store_slice(response_address)
116                 .store_coins(0)
117                 .store_msgbody_prefix_slice()
118                 .store_op(op::excesses)
119                 .store_query_id(query_id);
120             send_raw_message(msg.end_cell(), CARRY_REMAINING_GAS);
121         }
122     }
123     return ();
124 }
```

Suggestion:

It is recommended to add the "IGNORE_ERRORS" flag.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

POO-1 Centralization Risk

Severity: Medium

Status: Fixed

Code Location:

contracts/AMM/pool.fc#1

Descriptions:

Centralization risk was identified in the smart contract:

1. Admin can upgrade many contracts.
2. Admin can immediately update storage of many contracts.

Suggestion:

It is recommended to take measures to reduce the risk of centralization, such as an off-chain multi-signature mechanism for the admin.

Resolution:

This issue has been fixed. The client has a multi-signature mechanism for admin.

SWA-1 Potentially incorrect use of MIN_SWAP_AMOUNT in one of the checks

Severity: Medium

Status: Fixed

Code Location:

contracts/AMM/markets/swaps.fc#18

Descriptions:

Each `handle_swap_xx_for_xx()` function in this file checks for `jetton_amount > 0`:

```
throw_unless(error::not_enough_jettons, jetton_amount > 0);
```

However, in `handle_swap_sy_for_pt()` it checks:

```
throw_unless(error::not_enough_jettons, jetton_amount > MIN_SWAP_AMOUNT);
```

This breaks the symmetry between similar swap functions and may prevent certain valid swaps from executing.

Suggestion:

In `handle_swap_sy_for_pt()`, change the condition to `jetton_amount > 0` to align with other `handle_swap_xx_for_xx()` functions.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

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.

