TFM
Router IBC Contract
Audit Report

Prepared for TFM, 6th July 2023

Table of Contents

Introduction

SCV was engaged by TFM to assist in identifying security threats and vulnerabilities that have the potential to affect their security posture. Additionally, SCV will assist the team in understanding the risks and identifying potential mitigations.

Scope

SCV performed the security assessment on the following codebase:

- https://github.com/tfm-com/tfm_router_ibc
    - Code Freeze: `cebdff5e7add4344b4d77bcea0f2c0cb1c2108f3`

The TFM team implemented the suggestions and remediations outlined in this report, with SCV performing the revisions on the specified codebase and commit.

- https://github.com/tfm-com/tfm_router_ibc
    - Code Freeze: `aa057b7be32246d0ffb33925575d072c1c4f5143`

Methodologies

SCV performs a combination of automated and manual security testing based on the scope of testing. The testing performed is based on the extensive experience and knowledge of the auditor to provide the greatest coverage and value to TFM. Testing includes, but is not limited to, the following:

- Understanding the application and its code base purpose.
- Deploying SCV in-house tooling to automate dependency analysis and static code review.
- Analyze each line of the code base and inspect application security perimeter.
- Review underlying infrastructure technologies and supply chain security posture.

Code Criteria and Test Coverage

This section below represents how *SUFFICIENT* or *NOT SUFFICIENT* each code criteria was during the assessment

Criteria	Status	Notes
Provided Documentation	**SUFFICIENT**	N/A
Code Coverage Test	**SUFFICIENT**	While it is recommended to have a minimum test coverage of 40%, the current coverage stands at 37%. We consider it sufficient due to the inclusion of integration testing and extensive in-line comments. However, TFM should consider improving test coverage.
Code Readability	**SUFFICIENT**	The codebase had good readability and utilized many Rust and CosmWasm best practices.
Code Complexity	**SUFFICIENT**	N/A

Threat Modeling

The goal of threat modeling is to identify and evaluate potential threats to a system or application and to develop strategies to mitigate or manage those threats. Threat modeling is an important part of the software development life cycle, as it helps developers and security professionals to proactively identify and address security risks before they can be exploited by attackers.

The main objectives of threat modeling includes (not limited to) the following :

- **Identify threats**: The first objective of threat modeling is to identify potential threats that could affect the security posture of the underlying smart contracts or application. This can include threats from external attackers, internal actors, or even accidental events that could happen.
- **Evaluate risks:** Once potential threats have been identified, the next objective is to evaluate the risks associated with each threat. This involves assessing the likelihood of each threat occurring and the potential impact it could have overall.
- **Mitigation strategies:** After identifying potential threats and evaluating the associated risks, the next objective is to develop strategies to mitigate or reduce the impact of threats. This can include implementing technical controls, such as access controls or further security measures around developing policies and procedures to reduce the likelihood or impact of a threat.
- **Communicate findings:** The final objective of threat modeling is to communicate the findings and recommendations to relevant stakeholders, such as developers, security teams, and management. This helps ensure that everyone involved in the development and maintenance understands the potential risks and the best strategies for addressing them.

Vulnerabilities Summary

#	Summary Title	Risk Impact	Status
1	Invalid recovery address causes funds to get stuck in the contract	**Medium**	**Remediated**
2	`ExecuteSwapOperations` does not validate funds sent	**Medium**	**Remediated**
3	Permissionless `SendBalance` message allows attackers to drain available funds	**Medium**	**Remediated**
4	Smart queries are not enabled for `INFLIGHT_PACKETS` storage	**Low**	**Remediated**
5	Migrate function does not update the contract's name and version	**Informational**	**Remediated**
6	Redundant error check when parsing IBC replies	**Informational**	**Remediated**
7	Unnecessary lowercase address validation	**Informational**	**Remediated**
8	General codebase inefficiencies	**Informational**	**Partially Remediated**
9	Owner cannot drop pending ownership	**Informational**	**Remediated**

Detailed Vulnerabilities

1 – Invalid recovery address causes funds to get stuck in the contract

---

Risk Impact: Medium - **Status**: Remediated

Description

The recovery address contained inside the `SwapIbcHookMsg` struct is not validated when used in `contracts/router/src/executors.rs:227`. In case when an IBC acknowledgment error or timeout occurs, a bank message is executed to refund the funds in the `receive_ack` and `receive_timeout` functions.

If the recovery address is invalid, the transaction will fail, and the funds will not be refunded. An attacker can steal them by executing the `SendBalance` function with the `asset_info` parameter as the leftover asset and the `to` parameter as their own address.

Recommendations

Consider validating the recovery address to ensure it is valid so refunding will not fail.

2 – `ExecuteSwapOperations` does not validate funds sent

Risk Impact: Medium - **Status**: Remediated

Description

The `ExecuteSwapOperations` messages in `contracts/router/src/executors.rs:53` does not check that sent funds equal the sum of all offer amounts in the swap routes with `start_asset` as denom. The current implementation only ensures the sent funds are not empty. As a result, funds that were previously stored in the contract could be drained by sending a different denom with the `start_asset` as a different denom.

While the contract is not designed to hold any funds at present, it is crucial to address potential bugs that could inadvertently allow funds to accumulate within the contract.

Recommendations

Consider validating the sent funds' denom equal to `offer_asset_info` while the amount equal to the sum of all `offer_amount` in the `SwapRoute` struct.

3 – Permissionless `SendBalance` message allows attackers to drain available funds

Risk Impact: Medium - **Status**: Remediated

Description

The `send_balance` function in `contracts/router/src/executor.rs:166` doesn't ensure that the `info.sender` is `env.contract.address`. This allows anyone to call the message to transfer any funds present in the contract.

Similar to the #2 vulnerability, it is crucial to address potential bugs that could inadvertently allow funds to accumulate within the contract.

Recommendations

Consider validating the sender is the contract itself.

4 – Smart queries are not enabled for `INFLIGHT_PACKETS` storage

Risk Impact: Low - **Status**: Remediated

Description

The `INFLIGHT_PACKETS` storage state is used to record the transfer packet information such as the recovery address, channel, sequence identifier, and denom transmitted. This information is helpful for users to know the current status of their IBC packet transfers. However, it is not exposed in query messages.

Recommendations

Consider exposing a query entry point to allow users to query `INFLIGHT_PACKETS` storage values.

5 – Migrate function does not update the contract's name and version

Risk Impact: Informational - **Status**: Remediated

Description

The migrate function does not call the `set_contract_version` function to update the contract name and version to the latest values. In a contract migration, the contract will still record the old contract name and version, misleading users.

Recommendations

Consider calling the `set_contract_version` function in the migrate entry point.

6 – Redundant error check when parsing IBC replies

Risk Impact: Informational - **Status**: Remediated

Description

The `handle_forward_reply` function tries to parse the result after sending the IBC transfer packet in `contracts/router/src/executors.rs:259-261`. This will never be entered because the sub-message is executed as `reply_on_success`, indicating that the reply handler will only execute if the transaction succeeded.

Recommendations

Consider removing the unneeded lines to reduce gas consumption.

7 – Unnecessary lowercase address validation

Risk Impact: Informational - **Status**: Remediated

Description

The codebase utilizes `addr_validate_to_lower` to validate the addresses' validity by converting them into lowercase letters before calling `addr_validate`. The lowercase conversion is unnecessary because the `addr_validate` function will automatically ensure the address is lowercase and errors if not.

Recommendations

Consider removing the lowercase conversion and directly calling `addr_validate` instead.

8 – General codebase inefficiencies

Risk Impact: Informational - **Status**: Partially Remediated

Description

The following code lines contain unused code:

- `contracts/router/src/contract.rs:132-140` will be unused because there is no `MsgReplyID::Swap` configured.
- `packages/tfm/src/router.rs:159`

The following code lines contain unused imports:

- `contracts/router/src/utils.rs:6:5`
- `contracts/router/src/utils.rs:4:20`
- `contracts/router/src/utils.rs:8:5`

A TODO comment is found in `contracts/router/src/executors.rs:52`. And lastly, `contracts/router/src/executors.rs:281` contains an incorrect comment.

Recommendations

Consider removing the unused code lines, unused imports, resolving TODO comments, and updating the incorrect comments accordingly.

9 – Owner cannot drop pending ownership

Risk Impact: Informational - **Status**: Remediated

Description

The contract allows setting a new owner and accepting the pending ownership via the `set_owner` and `accept_ownership` functions. In a scenario where a new proposed owner is set up and needs to be canceled, the owner couldn't reset the proposed owner back to an empty string, similar to how the `instantiate` function does this in `contracts/router/src/contract.rs:51`.

Recommendations

Consider exposing a new entry point that allows the current owner to remove the proposed owner.

Document control

Version	Date	Approved by	Changes
-	1/07/2023	-	Audit Commencement
0.1	05/07/2023	Vinicius Marino	Document Pre-Release
0.2	06/07/2023	SCV Team	Remediation Revisions
1.0	06/07/2023	Vinicius Marino	Document Release

Appendices

A. Appendix – Risk assessment methodology

A qualitative risk assessment is performed on each vulnerability to determine the impact and likelihood of each.

Risk rate will be calculated on a scale. As per criteria Likelihood vs Impact table below:

	Rare	Unlikely	Possible	Likely
Critical	**Medium**	**Severe**	**Critical**	**Critical**
Severe	**Low**	**Medium**	**Severe**	**Severe**
Moderate	**Low**	**Medium**	**Medium**	**Severe**
Low	**Low**	**Low**	**Low**	**Medium**
Informational	**Informational**	**Informational**	**Informational**	**Informational**

LIKELIHOOD

- Likely: likely a security incident will occur;
- Possible: It is possible a security incident can occur;
- Unlikely: Low probability a security incident will occur;
- Rare: In rare situations, a security incident can occur;

IMPACT

- Critical: May cause a significant and critical impact;
- Severe: May cause a severe impact;
- Moderate: May cause a moderated impact;
- Low: May cause low or none impact;
- Informational: May cause very low impact or none.

B. Appendix – Report Disclaimer

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts SCV-Security to perform a security review. The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only. The content of this audit report is provided “as is”, without representations and warranties of any kind, and SCV-Security disclaims any liability for damage arising out of, or in connection with, this audit report.