AUDIT REPORT

TerraForm Labs Inscription Bridge

Prepared by SCV-Security

On 8th April 2024

Table of Contents

Introduction

SCV has been engaged by TerraForm Labs to conduct a comprehensive security review with the goal of identifying potential security threats and vulnerabilities within the codebase. The purpose of this audit is to evaluate the security posture of the codebase and provide actionable recommendations to mitigate any identified risks. This report presents an overview of the findings from our security audit, outlining areas of concern and proposing effective measures to enhance the codebase's security.

Scope Functionality

The Inscription Bridge is a Go script that allows users to move Cosmos CFT-20 Token Inscriptions between the Cosmos Hub and Terra. The script analyses and stores blocks and transactions on the Cosmos Hub and Terra in a database and verifies valid inscriptions with the Asteroid Protocol API endpoint.

When a user sends an inscription to the bridge on the Cosmos Hub, the inscription is locked on the hub and a new token is minted on Terra. Then, when the user sends the bridged token back to the bridge on Terra, the minted token is burned and the inscription is sent back to the user on the Cosmos Hub.

Submitted Codebase

locked-astroport-vault	
Repository	https://github.com/BIG-Labs/inscription-bridge
Commit	`7088f0988763597e77c2c04eefef4a037a000623`
Branch	`main`

Revisions Codebase

locked-astroport-vault	
Repository	https://github.com/BIG-Labs/inscription-bridge
Commit	`c6c54409b7b34c8a6e4762eb07644c3c8fc8f4e6`
Branch	`main`

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 TerraForm Labs. Testing includes, but is not limited to, the following:

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

Code Criteria

This section provides an evaluation of specific criteria aspects as described below:

- **Documentation:** Evaluating the presence and comprehensiveness of publicly available or provided explanatory information, diagram flowcharts, comments, and supporting documents to enhance code understanding.
- **Coverage:** Evaluating whether the code adequately addresses all necessary cases and scenarios, ensuring that the intended functionality or requirements are sufficiently covered.
- **Readability:** Assessing how easily the code can be understood and maintained, considering factors such as code structure, naming conventions, and overall organisation.
- **Complexity:** Evaluating the complexity of the code, including factors such as, number of lines, conditional statements, and nested structures.

The status of each criteria is categorised as either **SUFFICIENT** or **NOT-SUFFICIENT** based on the audit assessment. This categorisation provides insights to identify areas that may require further attention and improvement.

Criteria	Status	Notes
Documentation	**SUFFICIENT**	N/A
Coverage	**SUFFICIENT**	Testing coverage is considered sufficient, although there is room for improvement as the existing test cases can be extended and contain some todos.
Readability	**SUFFICIENT**	The codebase had good readability overall and utilised many Go best practices.
Complexity	**SUFFICIENT**	N/A

Findings Summary

Summary Title	Risk Impact	Status
Attacker spam database with invalid messages	MODERATE	RESOLVED
Ensure tokenfactory denom is created before starting bridge	MODERATE	ACKNOWLEDGED
Safety measures can be added to the protocol	MODERATE	ACKNOWLEDGED
Remove hard coded gas configs	INFO	RESOLVED
Misleading error messages may be logged while submitting transactions	INFO	RESOLVED
Remove unused config variables	INFO	RESOLVED
Remove unused functions	INFO	RESOLVED
Update incorrect message url	INFO	RESOLVED

Audit Observations

The audit observations section is intended to present potential findings that are related to the underlying design of the protocol and would require underlying design changes to remediate that may change the overall functioning of the protocol. SCV asks that the client formulate responses to add context to validate or invalidate the following concerns.

1. Hybrid contract architecture can provide more robust security guarantees

While the current architecture does elegantly provide a simple and effective solution for managing inscription bridging all through a single backend application a more robust approach could be considered in the future. By implementing a contract on the Terra side of the bridge it will allow for more of the logic to be brought on-chain and will provide better error handling, atomicity guarantees, and more trustlessness and transparency. We recommend investigating this approach for future versions of the protocol.

Revision Notes

The client acknowledged that the hybrid contract approach would be a better approach for future versions of the inscription bridge, however they do not plan on implementing it in the current iteration.

2. Asteroid Protocol is unaudited

Asteroid Protocol is unaudited and contains the following quotation at the end of its GitHub README: *“Remember, all software and specifications are experimental, unaudited and not "endorsed" by any entity including Delphi Labs and the Astroport Foundation. Use at your own risk.”* Since the bridge relies on the Asteroid Protocol inscription metaprotocol, indexer, and API endpoint, the team should continuously follow the protocol to ensure that no breaking changes are introduced. Additionally, if Asteroid Protocol is audited then the inscription

bridge should be re-evaluated as well for potential vulnerabilities that are identified with the protocol.

3. Remove hard-code database credentials and mnemonic

The codebase submitted for review contains hard-coded database credentials and a hard coded mnemonic phrase. This is marked as an audit observation because the team states that these values will be removed before the code is released in production.

Findings Technical Details

1. Attacker spam database with invalid messages

RISK IMPACT: MODERATE	STATUS: RESOLVED

Revision Notes

The client has updated the fee amount which will raise the cost of a spam attack considerably. This revision does not completely mitigate the risk of a spam attack but it will considerably decrease the likelihood.

Description

The Terra bridge's `VerifySendMsg` function in `internal/bridge/bridge.go:118` analyses and filters transactions in the block to determine if they are inscription bridging transactions from Terra to the Cosmos Hub. This filtering and validation is done to ensure that only bridge transactions are processed further.

The actual amount that is bridged is directly pulled from the memo of the message. There is nothing that prevents an attacker from spamming many valid `/cosmos.bank.v1beta1.MsgSend` messages with modified memo amount that pass all the bridge validations but will error when the bridge attempts to perform a burn from that account because the account does not have enough of the denom to burn. This will spam the database with many `transaction_error` entries that may be difficult to filter.

Recommendation

We recommend updating the logic in the Terra bridge to more granularly detect the case where a user attempts to bridge tokens in excess of their balance. These errors can then be entered into the database in a separate index for likely spam errors.

2. Ensure tokenfactory denom is created before starting bridge

RISK IMPACT: MODERATE	STATUS: ACKNOWLEDGED

Revision Notes

The client has decided to manually verify this condition and has decided to not remediate the issue in the current version of the inscription bridge.

Description

The `TICKER_WHITELISTED` map contains the Terra tokenfactory denom to be minted when an inscription is bridged. Currently there is no logic to ensure that the tokenfactory denom is actually instantiated. If the bridge is started before the denom is instantiated, users can attempt to bridge inscriptions, but they will error when `SubmitMintTransaction` is called which will effectively lock inscriptions in the bridge unless they can be manually forwarded.

Recommendation

We recommend adding a query to ensure the tokenfactory denom has been created when the bridge program is started before proceeding with the setup and processing blocks. The bridge can query the tokenfactory denom metadata to ensure a valid response is returned before proceeding.

3. Safety measures can be added to the protocol

RISK IMPACT: MODERATE	STATUS: ACKNOWLEDGED

Revision Notes

The client acknowledged this concern and is not making any updates with the current iteration of the inscription bridge. However, they are looking at improving the error handling cases for when users interact with the bridge while the Cosmos Hub or Terra is either updating or unresponsive in a future iteration.

Description

The inscription bridge should consider implementing invariants that ensure that the protocol parameters fall within the expected values. These invariants can be very important if the protocol has a bug or is actively being exploited.

The most important invariant to check is to ensure that the amount of tokenfactory denom is always equal to the amount of its related inscription that is held in the cosmos bridge address.

While this does not prevent users from manually creating their own transactions, the frontend could also provide logic to block bridging activity in times when the invariant conditions are not met.

While the nature of inscription indexing does not allow the security guarantees provided by atomic on-chain transactions, it is important to implement security checks throughout the inscription bridge process.

Recommendation

We recommend adding invariant conditions to the inscription bridge and enforcing them in both the frontend and the bridge program.

4. Remove hard coded gas configs

RISK IMPACT: INFO	STATUS: RESOLVED

Description

The `SubmitTransaction` function in `internal/blockchain/transaction_provider.go:109-114` includes hard coded gas adjustments. It is best practice to create a gas adjustment variable and add code comments to explain the values.

Recommendation

We recommend creating gas adjustment variables and adding code comments to explain their values.

5. Misleading error messages may be logged while submitting transactions

RISK IMPACT: INFO	STATUS: RESOLVED

Description

The bridge keeps track of any errors that occur while processing blocks and transactions and when creating and sending transactions. Error structs containing information about the error state and error message are then stored in the DB for manual review by the inscription bridge team.

However, the following error messages are swallowed and will eventually be stored in the DB as messages that don't accurately represent the error state:

- `/internal/blockchain/transaction_provider.go:31`
- `/internal/blockchain/transaction_provider.go:227`
- `/internal/blockchain/transaction_provider.go:228`
- `/internal/blockchain/transaction_provider.go:242`

Recommendation

We recommend adding error checks for all potential error conditions to accurately represent the error state in the DB.

6. Remove unused config variables

RISK IMPACT: INFO	STATUS: RESOLVED

Description

The `COSMOS_WEBSOCKET`, `HUB_FEE_AMOUNT`, and `TERRA_FEE_AMOUNT` config variables in `config/config.go` are not used. It is best practice to remove any unused config variables.

Recommendation

We recommend removing the unused config variables from `config.go`.

7. Remove unused functions

RISK IMPACT: INFO	STATUS: RESOLVED

Description

The following functions are currently unused:

- `IsTicWhitelisted` in `internal/bridge/bridge.go:218`
- `SubmitCreateFactoryTransaction` in `internal/blockchain/transaction_provider.go:241`
- `VerifyBurnMsg` in `internal/bridge/bridge.go:143`

Recommendation

We recommend removing the unused functions from the codebase.

8. Update incorrect message url

RISK IMPACT: INFO	STATUS: RESOLVED

Description

The `VerifyBurnMsgfunction` in `internal/bridge/bridge.go:143` checks for the message url `"/osmosis.tokenfactory.v1beta1.MsgBurn"`. While the `VerifyBurnMsgfunction` is currently unused, if it is intended to be used the message url should be updated.

Recommendation

We recommend updating the message url in `VerifyBurnMsgfunction`.

Document Control

Version	Date	Notes
-	18th March 2024	Security audit commencement date.
0.1	26th March 2024	Initial report with identified findings delivered.
0.5	28th March - 6th April	Fixes remediations implemented and reviewed.
1.0	8th April 2024	Audit completed, final report delivered.

Appendices

A. Appendix – Risk assessment methodology

SCV-Security employs a risk assessment methodology to evaluate vulnerabilities and identified issues. This approach involves the analysis of both the LIKELIHOOD of a security incident occurring and the potential IMPACT if such an incident were to happen. For each vulnerability, SCV-Security calculates a risk level on a scale of 5 to 1, where 5 denotes the highest likelihood or impact. Consequently, an overall risk level is derived from combining these two factors, resulting in a value from 10 to 1, with 10 signifying the most elevated level of security risk

Risk Level	Range
CRITICAL	10
SEVERE	From 9 to 8
MODERATE	From 7 to 6
LOW	From 5 to 4
INFORMATIONAL	From 3 to 1

LIKELIHOOD and **IMPACT** would be individually assessed based on the below:

Rate	LIKELIHOOD	IMPACT
5	**Extremely Likely**	Could result in severe and irreparable consequences.
4	**Likely**	May lead to substantial impact or loss.
3	**Possible**	Could cause partial impact or loss on a wide scale.
2	**Unlikely**	Might cause temporary disruptions or losses.
1	**Rare**	Could have minimal or negligible impact.

B. Appendix – Report Disclaimer

This report should not be regarded as an "endorsement" or "disapproval" of any specific project or team. These reports do not indicate the economics or value of any "product" or "asset" created by a team or project that engages SCV-Security for a security review. The audit report does not make any statements or warranties about the code's utility, safety, suitability of the business model, regulatory compliance of the business model, or any other claims regarding the fitness of the implementation for its purpose or its bug-free status. The audit documentation is intended 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 damages arising from or in connection with this audit report. Copyright of this report remains with SCV-Security.

THANK YOU FOR CHOOSING

SCV SECURITY

scv.services

contact@scv.services