Bifrost TerraChain Integration Report

Prepared for Thorchain, 17 March 2022

Table of Contents

Document control

Document changes

Version	Date	Name	Changes
0.1	2022-03-10	Vinicius Marino	Initial report
0.2	2022-03-10	Joshua Padman	Report updates
0.3	2022-03-11	Vinicius Marino	Report review
0.4	2022-03-11	Joshua Padman	Report review
0.5	2022-03-11	Vinicius Marino	Team communication and Pre-Release
1.0	2022-03-17	Vinicius Marino	Final report release

Document contributors

Name	Role	Email address
Vinicius Marino	Security Specialist	vini@scv.services
Joshua Padman	Security Specialist	josh@scv.services

Introduction

SCV was engaged by Thorchain 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 Thorchain Gitlab PR:

- https://gitlab.com/thorchain/thornode/-/merge_requests/2077

For reference, the SHA hash related to the PR was *fb8c9ac71adaa3c6497b4a2d61eacdf261fa6017*.

The scope was limited to the integration between `Bifrost` and `TerraChain` along it's implementation connecting both chains. Additionally, edge testing cases concerns were raised by Thorchain team in a form of questions and addressed individually by SCV.

`Bifrost` enables multi-chain connectivity by observing activity from chains and processing transactions between blockchains.

Vulnerabilities were remediated by Thorchain team in the following merge request:

- https://gitlab.com/thorchain/thornode/-/merge_requests/2165

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 Thorchain. 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;
- Analysis of each line of the code base and inspect application perimeter;
- Review underlying infrastructure technologies and supply chain security posture;

Code Criteria and Test Coverage

SCV is using a scale from **0** to **10** that represents how `SUFFICIENT(6-10)` or `NOT SUFFICIENT(0-5)` each code criteria was assessed:

Criteria	Status	Scale Range	Notes
Provided Documentation	**Sufficient**	7-8	N\A
Code Coverage Test	**Sufficient**	6-8	N\A
Code Readability	**Sufficient**	7-8	N\A
Code Complexity	**Sufficient**	7-8	N\A

Vulnerabilities Summary

	Title and Summary	Risk	Status
1	gRPC not using TLS & connecting WithInsecure()	**Low**	**Acknowledged**
2	Potential division by zero case can cause a Panic	**Low**	**Remediated**
3	Txs order might not be the same between API endpoints	**Low**	**Acknowledged**
4	Improving code quality in common.md	**Informational**	**Remediated**
5	Incorrect logging level/confusing error description in cosmos_client.go	**Informational**	**Remediated**
6	Missing additional test cases on common files	**Informational**	**Acknowledged**
7	Terra native demon might not be fully supported when paying fees	**Informational**	**Acknowledged**
8	Unnecessary code adds complexity to the code base	**Informational**	**Acknowledged**
9	Unnecessary code in the cosmos_block_scanner.go	**Informational**	**Remediated**
10	UST denom not defined Thorchain assets.	**Informational**	**Acknowledged**

Detailed Vulnerabilities

Vulnerability 1: gRPC not using TLS & connecting WithInsecure()

Likelihood	Impact	Risk
Rare	Moderate	**Low**

Notes

Team suggests that, bifrost communicates with chain clients within a K8s cluster that is not exposed to the public internet.

Description

The Terra chainclient connects to the node using grpc and uses the option `grpc.WithInsecure()`. By defining this option it removes any transport layer security checks. This could allow an attacker to modify data being observed by the block scanner.

Common deployment architecture and node consensus for observed transactions make the likelihood of this being exploited rare.

Recommendations

Consider implementing TLS for gRPC communications.

Vulnerability 2: Potential division by zero case can cause a Panic

Likelihood	Impact	Risk
Rare	Low	**Low**

Description

The `averageFee` method divides by the length of the feeCache. Whilst very unlikely that averageFee would be called before the `feeCache` has a single entry, it could, and would lead to a division by zero error and consequential panic.

```
func (c *CosmosBlockScanner) averageFee() ctypes.Uint {
    sum := ctypes.NewUint(0)
    for _, val := range c.feeCache {
        sum = sum.Add(val)
    }
    return sum.Quo(ctypes.NewUint(uint64(len(c.feeCache))))
}
```

Recommendations

Ensure there is a check for the length of `c.feeCache` before performing arithmetic division operations.

Vulnerability 3: Txs order might not be the same between API endpoints

Likelihood	Impact	Risk
Rare	Moderate	**Low**

Notes

The entire Terra columbus-5 blocks were processed by SCV and a millions were processed by Thorchain team and confirmed that ordering is consistent. Only a hard fork and breaking change in Tendermint could change this behavior. It's current not possible to correlate the Tx body from GetBlock() to the Tx result in GetBlockResult().

Description

The merge request has a note about an assumption that the order of transactions from `rawTxs` (`getBlock`) is the same as the `TxsResults` (`blockResults`).

Tendermint documentation provides guidance that this assumption holds, as per link below:

- https://github.com/tendermint/tendermint/blob/v0.34.8/rpc/core/blocks.go#L130-L135

If the order is not equally matching between (`blockResults` and `TxsResults`) it could lead to impact data integrity.

Recommendations

It might be worth reaching out to the Tendermint developer team to ensure documentation is accurate. Alternatively, request every transaction individually from the API or use an indexer, if possible.

Vulnerability 4: Improving code quality in common.md

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

In `common/chain.go#L96-L102` there are a two small nit picks. The `GetSigningAlgo()` returns `SigningAlgoSecp256k1` regardless, as the switch has a single, default, case defined.

```
// GetSigningAlgo get the signing algorithm for the given chain
func (c Chain) GetSigningAlgo() SigninAlgo {
   switch c {
   default:
       return SigningAlgoSecp256k1
   }
}
```

Also, `Validate()` uses `if ch < 'A' || ch > 'Z'` to determine if all characters are capitals. Current implementation works, however, it can be replaced with `!unicode.IsUpper(ch)` to use the core libraries.

Recommendations

Consider making changes to decrease complexity and increase readability of code.

Vulnerability 5: Incorrect logging level/confusing error description in cosmos_client.go

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

When a whitelisted coin is unable to be converted this should probably be an error level log event. Especially when it is on outbound transactions (example `bifrost/pkg/chainclients/terra/cosmos_client.go#L308`).

Additionally, the error text is confusing, the only error return path from `fromThorToCosmos` says `"asset does not exist / not whitelisted by client"` and the logged error in `processOutboundTx` is `"wasn't able to convert coins that passed whitelist"`.

Recommendations

Consider changing the error text and increasing the logging level.

Vulnerability 6: Missing additional test cases on common files

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Notes

Thorchain team suggests that remediation would be addressed in a subsequent PR.

Description

Code coverage tests were missing from the following files:

- `common/address.go`
- `common/asset.go`
- `common/chain.go`
- `common/pubkey.go`

Recommendations

Consider adding new tests to cover the new functionality and values.

Vulnerability 7: Terra native demon might not be fully supported when paying fees

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Notes

Applicable for the TxOut only. The design of THORChain is such that gas are always paid in the L1 token of a chain. User experience is not impacted here.

Description

Not all Terra stable demon might be supported while using for fees. Terra allows fees to be paid using any stable and that could impact the user experience if only one asset `uluna` is taking into consideration.

```
    // only consider transactions with fee paid in uluna
    coin, err := fromCosmosToThorchain(fees[0])
    if err != nil || !coin.Asset.Equals(c.cfg.ChainID.GetGasAsset()) {
        return
    }
```

Recommendations

It's recommended to accept all Terra native when used for fee if possible

Vulnerability 8: Unnecessary code adds complexity to the code base

Likelihood	Impact	Risk
Likely	Informational	**Informational**

Notes

Thorchain team suggests that remediation would be addressed in a subsequent PR.

Description

In `bifrost/pkg/chainclients/loadchains.go` there are some sections of code that are unreachable or have no effect. These can lead to confusion whilst reading code.

This **`continue`** on `loadchains.go#90` will not be reached as the log call is a `Fatal()` and calls `.Msg()` which triggers the `os.Exit(1)` causing the program to terminate immediately.

The **`default`** on `loadchains.go#85` option has no code, it used to have a continue but even that was not necessary as it didn't skip any code.

Recommendations

Remove the **`continue`** from each case and remove the **`default`** option.

Vulnerability 9: Unnecessary code in the cosmos_block_scanner.go

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

The `default: continue` in the `cosmos_block_scanner.go#L340-L341` is not required if the switch isn't matched. Code has no effect.

Recommendations

Remove the unnecessary code.

Vulnerability 10: UST denom not defined Thorchain assets.

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Notes

This is needed since Thorchain wants to map TERRA-UST and not as TERRA-USD.

Description

The `cosmos_assets.go#L11-L22` has defined both denom for `uluna` and `uusd` from Terra and mapped in the Thorchain side as `LUNA` and `UST` respectively. However, only `LUNA` has been defined in `asset.go#L16`. This leads to all `uluna` and `uusd` denom transactions being scanned inconsistency.

Recommendations

Remove the definitions for `uusd` or properly add support to this denom.

Appendices

Appendix A: Report Disclaimer

THE CONTENT OF THIS AUDIT REPORT IS PROVIDED “AS IS”, WITHOUT REPRESENTATIONS AND WARRANTIES OF ANY KIND.

THE AUTHOR AND THEIR EMPLOYER DISCLAIM ANY LIABILITY FOR DAMAGE ARISING OUT OF, OR IN CONNECTION WITH, THIS AUDIT REPORT.

COPYRIGHT OF THIS REPORT REMAINS WITH THE AUTHOR.

Appendix B: 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:

Impact \ Likelihood	Rare	Unlikely	Possible	Likely
Critical	Medium	High	Critical	Critical
Severe	Low	Medium	High	High
Moderate	Low	Medium	Medium	High
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.


SCV