Sifchain - Margin

Cosmos Security Audit

Prepared by: **Halborn**

Date of Engagement: **July 5th, 2022 - July 29th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	07/15/2022	Chris Meistre
0.2	Document Edits	07/29/2022	John Saigle
0.3	Draft Review	07/29/2022	Gabi Urrutia
1.0	Remediation Plan	08/22/2022	Chris Meistre
1.1	Remediation Plan Updates	08/29/2022	Gokberk Gulgun
1.2	Remediation Plan Review	08/29/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Gokberk Gulgun	Halborn	Gokberk.Gulgun@halborn.com

Chris Meistre	Halborn	Chris.Meistre@halborn.com
John Saigle	Halborn	John.Saigle@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Sifchain engaged Halborn to conduct a security audit on their Margin module, beginning on July 5th, 2022 and ending on July 29th, 2022 . The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided nearly two weeks for the engagement and assigned two full-time security engineers to audit the security of the Margin module. The security engineers are blockchain and smart-contract security experts with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit to achieve the following:

- Ensure that Margin module functions are intended.
- Report potential security issues to the Sifchain Team.

In summary, Halborn identified few security risks that were accepted and addressed by the Sifchain Team.

It was found that several parameters relating to margin positions can be modified by a user with administrative privileges. These include values that affect interest liabilities and liquidation thresholds. These administrative privileges were linked to a unique private key and if this key were to get compromised, an attacker would have control over these parameters. After discussion with the Sifchain team, it was established that the key is held securely and a multi-signature key could be introduced in the future.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of the Margin module. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of structures and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture and purpose.
- Static Analysis of security for scoped repository, and imported functions. (staticcheck, gosec, unconvert, LGTM, ineffassign and semgrep)
- Manual Assessment for discovering security vulnerabilities on codebase.
- Ensuring correctness of the codebase.
- Dynamic Analysis on Margin module functions and data types.

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.

1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

IN-SCOPE:
The security assessment was scoped to Sifchain/sifnode repository.

Commit ID

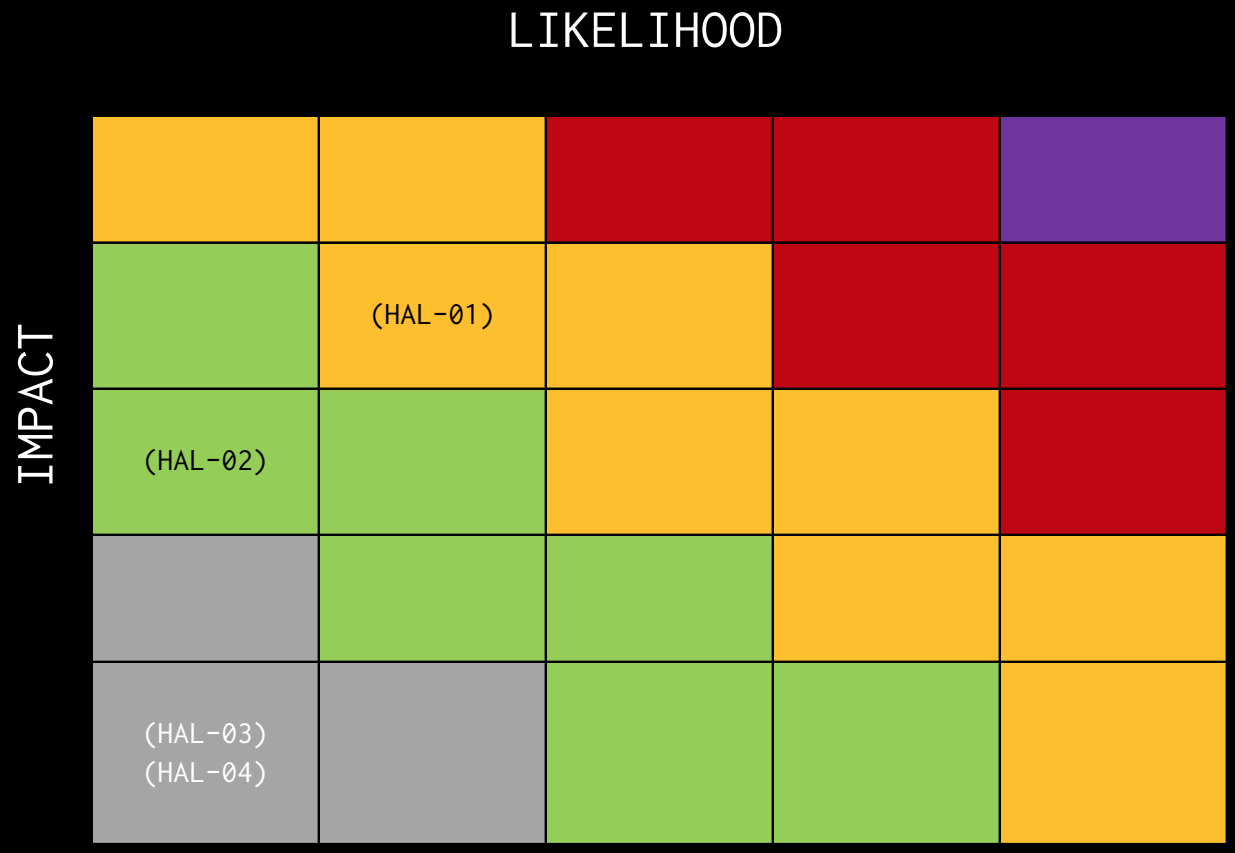
REMEDIATION COMMIT PROVIDED: Commit ID

IN-SCOPE Module:

- x/margin

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	1	1	2



EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL-01 - VALIDATION ERROR IN ALLOW-LIST FOR TOKENS THAT CAN BE USED AS COLLATERAL	Medium	SOLVED - 09/08/2022
HAL-02 - VULNERABLE 3RD PARTY PACKAGES	Low	SOLVED - 09/08/2022
HAL-03 - UNHANDLED ERRORS CAUSING PANIC	Low	SOLVED - 08/22/2022
HAL-04 - DUPLICATED ERROR CHECKS	Low	SOLVED - 08/22/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) VALIDATION ERROR IN ALLOW-LIST FOR TOKENS THAT CAN BE USED AS COLLATERAL - MEDIUM

Description:

The margin module allows users to open margin positions corresponding to a token pair in a liquidity pool. Users must deposit collateral to open a position. Pools consist of pairs of the native token rowan with an external asset.

Collateral is provided by depositing some amount of a supported token denomination into the pool. The native token rowan is always supported. A user with administrator privileges can determine which additional tokens are supported by enabling a setting on the corresponding liquidity pool.

It is possible for an attacker to create a counterfeit version of the native token rowan which can then be used as collateral for margin positions. Opening a margin position affects the price of assets within the liquidity pool. Therefore, an attacker with fake collateral can cause large price swings within the pools, which could lead to a drain of legitimate liquidity from the protocol. In this way, this attack is similar to a flash loan attack where an attacker takes out a large balance of rowan.

It is possible to perform this attack if an attacker can use the symbol Rowan as the denomination for their collateral. (Note: the capital 'R' distinguishes the counterfeit asset from the rowan token.)

This can be done in the case where a user can add a balance of a fake Rowan token to their account that meets the minimum threshold required to open a margin position, as well as a sufficient amount of rowan to pay transaction fees.

Example of correct validation: Using an invalid token “fakenotreal” to demonstrate that arbitrary tokens are rejected

Listing 1

```
$ sifnoded tx margin open --from user1 --keyring-backend test --
↳ borrow_asset ceth --collateral_asset fakenotreal --
↳ collateral_amount 100000000000000000000 --position long --fees
↳ 100000000000000000rowan --node tcp://localhost:26657 --chain-id
↳ testhub --broadcast-mode block --leverage 1 -y

[...]
raw_log: 'failed to execute message; message index: 0: fakenotreal
↳ : pool does not exist'
[...]
```

Creating a Rowan liquidity pool

Listing 2

```
# Opening a position
$ sifnoded tx margin open --from user1 --keyring-backend test --
↳ borrow_asset ceth --collateral_asset Rowan --collateral_amount
↳ 10000000000000000000 --position long --fees 100000000000000000
↳ rowan --node tcp://localhost:26657 --chain-id testhub --broadcast-
↳ mode block
[...]

# Querying open positions. The faked token `Rowan` appears as the
↳ collateral denomination.
logs:
$ sifnoded q margin positions-for-address
↳ sif1syavy2npfyt9tcncdtsdzf7kny9lh777yqc2nd

mtps:
- address: sif1syavy2npfyt9tcncdtsdzf7kny9lh777yqc2nd
  collateral_amount: "100000000000000000000"
  collateral_asset: Rowan
  custody_amount: "750000000000000000000"
  custody_asset: ceth
  id: "1"
  leverage: "1.000000000000000000"
  liabilities_i: "0"
```

```
  liabilities_p: "0"
  mtp_health: "0.597560975609756098"
  position: LONG
# At this stage, the allow-list has been subverted and 'Rowan' can
↳ be swapped for any other assets in the Sifchain liquidity pools
↳ as though it was an approved token
```

Code Location:

Several locations in the code base contain functionality that compare strings. When a **case-insensitive** comparison function is used (strings.EqualFold) Rowan can be considered equal to rowan. This allows Rowan to appear as a valid entry (as well as variations such as rOwan, roWan, etc.).

Recommendation:

Further validation should be performed on the denomination value used for collateral assets when opening margin positions. Providing any variation on the word “rowan” should give the same error message as the faketoken example above.

A previous Halborn audit of Sifchain’s CLP module contained a similar error that was fixed using a case-sensitive comparison. We recommend a review of the Sifnode code base be performed to identify and replace all uses of the EqualFold function with case-sensitive comparisons.

Remediation Plan:

SOLVED: The Sifchain Team has implemented a function, StringCompare, which is being used to replace the use of EqualFold throughout the code base.

The commit where the new function is located:

56c4e86302dec8c34c2fa4f435105e433a9a2e28.

The commit where the new function is being implemented:

b92b8e0498901ed34199b1bd88a1c6d9dea89d7b.

Risk Level:

Likelihood - 2
Impact - 4

3.2 (HAL-02) VULNERABLE 3RD PARTY PACKAGES - LOW

Description:

During the audit, Halborn identified some instances of installed 3rd party packages that contain known security vulnerabilities.

Packages:

ID	Package	Rating	Description
sonatype-2021-0598	tendermint	MEDIUM	Improper Input Validation
sonatype-2022-3945	go-buffer-pool	MEDIUM	Integer Overflow or Wraparound
CVE-2022-23327	go-ethereum	HIGH	Uncontrolled Resource Consumption
CVE-2022-21698	client_golang	HIGH	Denial of Service attack

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

We recommend that all 3rd party packages that are installed are kept up to date and all security fixes applied.

Remediation Plan:

SOLVED: The Sifchain Team has solved this by implementing the correct package versions.

The pull request where the packages are being upgraded:

3201.

3.3 (HAL-03) UNHANDLED ERRORS CAUSING PANIC - INFORMATIONAL

Description:

A variety of errors can occur during normal operation of the command-line client for the margin module. These errors lead to panics and/or major issues in user workflows:

Issue 1: There is an error that results in a panic because no validation is being performed on the `leverage` parameter when running the `tx margin open` command through the CLI.

Issue 2: The `force-close` command is not functional due to the code accessing a value `mtp_address` that is undefined.

Issue 3: In many cases, it is not possible for users to close margin positions that they have opened. Attempting to do so results in a panic.

Code Location:

Issue 1

```
❯ sifnoded tx margin open
panic: decimal string cannot be empty

goroutine 1 [running]:
github.com/cosmos/cosmos-sdk/types.MustNewDecFromStr(...)
        /home/ziion/go/pkg/mod/github.com/!sifchain/cosmos-sdk@v0.45.1-issue.11957/types/decimal.go:196
github.com/Sifchain/sifnode/x/margin/client/cli.GetOpenCmd.func1(0xc001048780?, {0x2f9aee8?, 0x0?, 0x0?})
        /home/ziion/Documents/SIFCHAIN/sifnode/x/margin/client/cli/tx.go:71 +0x57d
github.com/spf13/cobra.(*Command).execute(0xc001048780, {0x2f9aee8, 0x0, 0x0})
        /home/ziion/go/pkg/mod/github.com/spf13/cobra@v1.3.0/command.go:856 +0x67c
github.com/spf13/cobra.(*Command).ExecuteC(0xc000f86780)
        /home/ziion/go/pkg/mod/github.com/spf13/cobra@v1.3.0/command.go:974 +0x3b4
github.com/spf13/cobra.(*Command).Execute(...)
        /home/ziion/go/pkg/mod/github.com/spf13/cobra@v1.3.0/command.go:902
github.com/spf13/cobra.(*Command).ExecuteContext(...)
        /home/ziion/go/pkg/mod/github.com/spf13/cobra@v1.3.0/command.go:895
github.com/cosmos/cosmos-sdk/server/cmd.Execute(0x1?, {0xc000babfe0, 0x15})
        /home/ziion/go/pkg/mod/github.com/!sifchain/cosmos-sdk@v0.45.1-issue.11957/server/cmd/execute.go:36 +0x1eb
main.main()
        /home/ziion/Documents/SIFCHAIN/sifnode/cmd/sifnoded/main.go:18 +0x45
```

x/margin/client/cli/tx.go, Lines 67-71

Listing 3: (Line 71)

```
67              leverage, err := cmd.Flags().GetString("leverage")
68              if err != nil {
69                  return err
70              }
71              leverageDec := sdk.MustNewDecFromStr(leverage)
```

Issue 2

```
> /sifnoded tx margin force-close --id=1
Error: flag accessed but not defined: mtp_address
Usage:
  sifnoded tx margin force-close [flags]

Flags:
  -a, --account-number uint      The account number of the signing account (offline mode only)
  -b, --broadcast-mode string    Transaction broadcasting mode (sync|async|block) (default "sync")
      --dry-run                  ignore the --gas flag and perform a simulation of a transaction, but don't broadcast it
      --fee-account string       Fee account pays fees for the transaction instead of deducting from the signer
      --fees string              Fees to pay along with transaction; eg: 10uatom
      --from string              Name or address of private key with which to sign
      --gas string               gas limit to set per-transaction; set to "auto" to calculate sufficient gas automatically (
      --gas-adjustment float     adjustment factor to be multiplied against the estimate returned by the tx simulation; if t
      --gas-prices string        Gas prices in decimal format to determine the transaction fee (e.g. 0.1uatom)
      --generate-only            Build an unsigned transaction and write it to STDOUT (when enabled, the local Keybase is no
  -h, --help                     help for force-close
      --id uint                  id of the position
      --keyring-backend string   Select keyring's backend (os|file|kwallet|pass|test|memory) (default "os")
      --keyring-dir string       The client Keyring directory; if omitted, the default 'home' directory will be used
      --ledger                   Use a connected Ledger device
      --node string              <host>:<port> to tendermint rpc interface for this chain (default "tcp://localhost:26657")
      --note string              Note to add a description to the transaction (previously --memo)
      --offline                  Offline mode (does not allow any online functionality
  -o, --output string            Output format (text|json) (default "json")
  -s, --sequence uint            The sequence number of the signing account (offline mode only)
      --sign-mode string         Choose sign mode (direct|amino-json), this is an advanced feature
      --timeout-height uint      Set a block timeout height to prevent the tx from being committed past a certain height
  -y, --yes                      Skip tx broadcasting prompt confirmation

Global Flags:
      --chain-id string     The network chain ID
      --home string         directory for config and data (default "/Users/user/.sifnoded")
      --log_format string   The logging format (json|plain) (default "plain")
      --log_level string    The logging level (trace|debug|info|warn|error|fatal|panic) (default "info")
      --trace               print out full stack trace on errors
```

Figure 1: Force close fails due to undefined flag

x/margin/client/cli/tx.go, Lines 67-71

Listing 4: (Line 144)

```
144             MtpAddress, err := cmd.Flags().GetString("mtp_address"
 ↳ )
145             if err != nil {
146                 return err
147             }
```

Issue 3

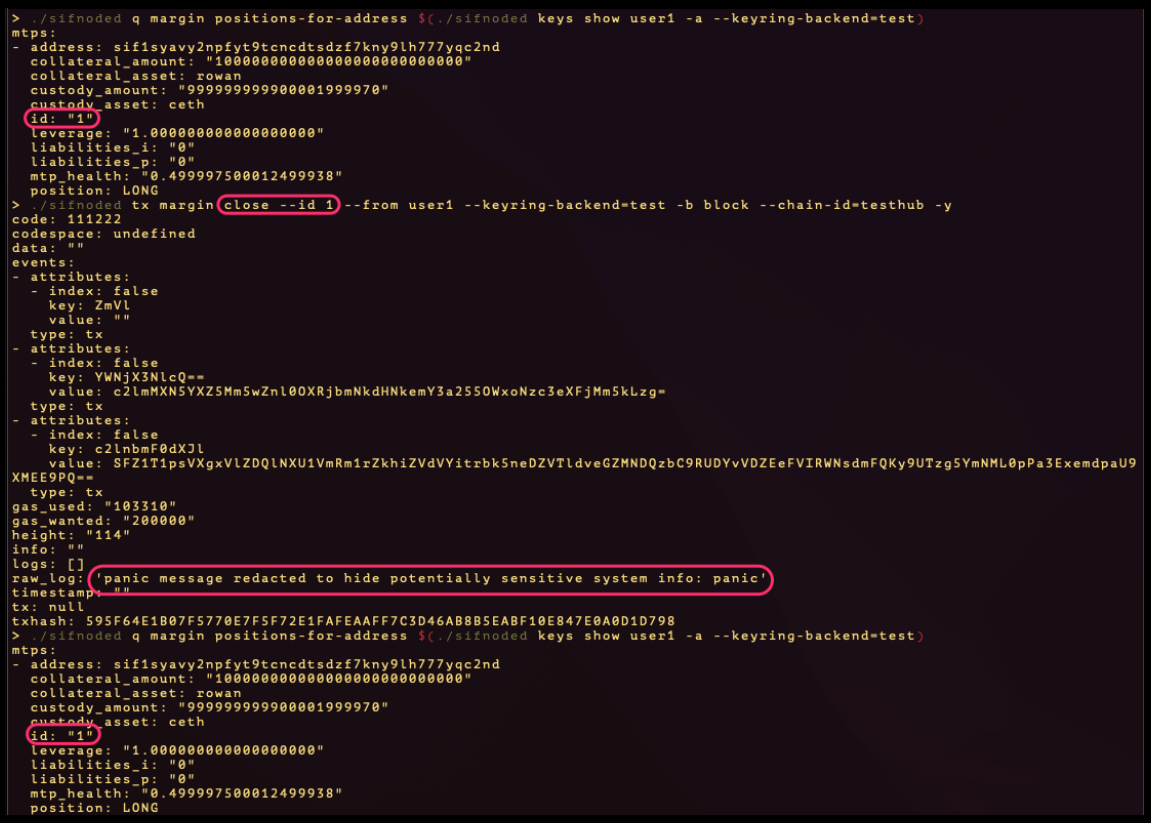


Figure 2: Panic when attempting to close position. After the error message occurs, the position remains open

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended that there is validation implemented for the leverage parameter to prevent any crashes. Additionally, the check for mtp_addresss should be fully implemented or fully removed depending on the development roadmap to enable the force-close action.

Extensive testing on command-line parameters should be performed before deployment. The use of the Simulations feature in Cosmos can assist with finding bugs that arise from unexpected input.

Remediation Plan:

SOLVED: The code was updated in commit 5e24b0efcb7a0cd881bdabbec3be988ffba90387.

3.4 (HAL-04) DUPLICATED ERROR CHECKS - INFORMATIONAL

Description:

There are two instances where an error check is not required, and the logic can be adjusted to only return the value.

Code Location:

x/margin/client/cli/tx.go, Lines 82-87

Listing 5: (Line 83)

```
82          err = tx.GenerateOrBroadcastTxCLI(clientCtx, cmd.Flags(),
↳ &msg)
83          if err != nil {
84              return err
85          }
86
87          return nil
```

x/margin/client/cli/tx.go, Lines 120-125

Listing 6: (Line 121)

```
120         err = tx.GenerateOrBroadcastTxCLI(clientCtx, cmd.Flags(),
↳ &msg)
121         if err != nil {
122             return err
123         }
124
125         return nil
```

x/margin/client/cli/tx.go, Lines 160-165

Listing 7: (Line 161)

```
160         err = tx.GenerateOrBroadcastTxCLI(clientCtx, cmd.Flags(),
 ↳ &msg)
161         if err != nil {
162             return err
163         }
164
165         return nil
```

x/margin/client/cli/tx.go, Lines 199-204

Listing 8: (Line 200)

```
199             err = tx.GenerateOrBroadcastTxCLI(clientCtx, cmd.Flags
 ↳ (), &msg)
200             if err != nil {
201                 return err
202             }
203
204             return nil
```

x/margin/client/cli/tx.go, Lines 252-257

Listing 9: (Line 253)

```
252             err = tx.GenerateOrBroadcastTxCLI(clientCtx, cmd.Flags
 ↳ (), &msg)
253             if err != nil {
254                 return err
255             }
256
257             return nil
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

As the `err` variable will already be `nil` if no error has been generated by the function, the second `if err != nil` can be removed.

Remediation Plan:

SOLVED: The code was updated in commit `5e24b0efcb7a0cd881bdabbec3be988ffba90387`.

AUTOMATED TESTING

Description:

Halborn used automated testing techniques to enhance coverage of certain areas of the scoped component. Among the tools used were staticcheck, gosec, semgrep, unconvert, LGTM and Nancy. After Halborn verified all the contracts and scoped structures in the repository and was able to compile them correctly, these tools were leveraged on scoped structures. With these tools, Halborn can statically verify security related issues across the entire codebase.

Semgrep - Security Analysis Output Sample:

Listing 10: Rule Set

```
semgrep --config "p/dgryski.semgrep-go" x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o dgryski.semgrep
semgrep --config "p/owasp-top-ten"      x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o owasp-top-ten.semgrep
semgrep --config "p/r2c-security-audit" x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o r2c-security-audit.semgrep
semgrep --config "p/r2c-ci"             x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o r2c-ci.semgrep
semgrep --config "p/ci"                 x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o ci.semgrep
semgrep --config "p/golang"             x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o golang.semgrep
semgrep --config "p/trailofbits"        x --exclude='*_test.go' --max-lines-per-finding 1000 --no-git-ignore -o trailofbits.semgrep
```

Semgrep Results:

Listing 11

```
Findings:

  clp/keeper/executors.go
     dgryski.semgrep-go.errnilcheck.err-nil-check
        superfluous nil err check before return
```

```
6         Details: https://sg.run/5Qd6
7
8         202 if err != nil {
9         203     return err
10        204 }
11        205 return nil
12          ---------------------------------------
13        227 if err != nil {
14        228     return err
15        229 }
16        230 return nil
17          ---------------------------------------
18      trailofbits.go.invalid-usage-of-modified-variable.invalid-
↳ usage-of-modified-variable
19        Variable `lp` is likely modified and later used on error.
↳ In some cases this could result
20        in panics due to a nil dereference
21        Details: https://sg.run/WWQ2
22
23        106 lp, err := k.GetLiquidityProvider(ctx, msg.
↳ ExternalAsset.Symbol, msg.Signer)
24        107 if err != nil {
25        108     lp = k.CreateLiquidityProvider(ctx, msg.
↳ ExternalAsset, lpUnits, addr)
26        109     ctx.EventManager().EmitEvents(sdk.Events{
27        110         sdk.NewEvent(
28        111             types.EventTypeCreateLiquidityProvider,
29        112             sdk.NewAttribute(types.
↳ AttributeKeyLiquidityProvider, lp.String()),
30        113             sdk.NewAttribute(types.AttributeKeyHeight,
↳ strconv.FormatInt(ctx.BlockHeight(), 10)),
31        114         ),
32        115     })
33        116     lpUnits = sdk.ZeroUint()
34        117 }
35
36
37    clp/types/msgs.go
38      dgryski.semgrep-go.errnilcheck.err-nil-check
39        superfluous nil err check before return
40        Details: https://sg.run/5Qd6
41
42        219 if err != nil {
43        220     return err
```

```
44          221 }
45          222 return nil
46
47
48    clp/types/querier.pb.gw.go
49        trailofbits.go.questionable-assignment.questionable-
↳ assignment
50          Should `protoReq` be modified when an error could be
↳ returned?
51          Details: https://sg.run/qq6y
52
53           52 protoReq.Symbol, err = runtime.String(val)
54             ----------------------------------------
55           79 protoReq.Symbol, err = runtime.String(val)
56             ----------------------------------------
57          142 protoReq.Symbol, err = runtime.String(val)
58             ----------------------------------------
59          153 protoReq.LpAddress, err = runtime.String(val)
60             ----------------------------------------
61          180 protoReq.Symbol, err = runtime.String(val)
62             ----------------------------------------
63          191 protoReq.LpAddress, err = runtime.String(val)
64             ----------------------------------------
65          222 protoReq.LpAddress, err = runtime.String(val)
66             ----------------------------------------
67          256 protoReq.LpAddress, err = runtime.String(val)
68             ----------------------------------------
69          294 protoReq.LpAddress, err = runtime.String(val)
70             ----------------------------------------
71          328 protoReq.LpAddress, err = runtime.String(val)
72             ----------------------------------------
73          402 protoReq.Symbol, err = runtime.String(val)
74             ----------------------------------------
75          436 protoReq.Symbol, err = runtime.String(val)
76
77
78    dispensation/client/cli/tx.go
79        trailofbits.go.invalid-usage-of-modified-variable.invalid-
↳ usage-of-modified-variable
80          Variable `dispensationCount` is likely modified and later
↳ used on error. In some cases this
81          could result  in panics due to a nil dereference
82          Details: https://sg.run/WWQ2
83
```

AUTOMATED TESTING

```
84          117 dispensationCount, err := strconv.ParseInt(args[2],
↳ 10, 64)
85          118 if err != nil {
86          119     return fmt.Errorf("invalid dispensation count :%d",
↳  dispensationCount)
87          120 }
88
89
90   margin/client/cli/tx.go
91      dgryski.semgrep-go.errnilcheck.err-nil-check
92          superfluous nil err check before return
93          Details: https://sg.run/5Qd6
94
95           85 if err != nil {
96           86     return err
97           87 }
98           88
99           89 return nil
100            ----------------------------------------
101         123 if err != nil {
102         124     return err
103         125 }
104         126
105         127 return nil
106           ----------------------------------------
107         163 if err != nil {
108         164     return err
109         165 }
110         166
111         167 return nil
112           ----------------------------------------
113         202 if err != nil {
114         203     return err
115         204 }
116         205
117         206 return nil
118           ----------------------------------------
119         255 if err != nil {
120         256     return err
121         257 }
122         258
123         259 return nil
124
125
```

```
126   margin/keeper/keeper.go
127      trailofbits.go.questionable-assignment.questionable-
 ↳ assignment
128         Should `mtp` be modified when an error could be returned?
129         Details: https://sg.run/qq6y
130
131         472 mtp.MtpHealth, err = k.UpdateMTPHealth(ctx, *mtp, pool
 ↳ )%
```

Gosec - Security Analysis Output Sample:

Listing 12

```
[x/ethbridge/test/test_helpers.go:238] - G404 (CWE-338): Use of weak random number generator (math/rand instead of crypto/rand) (Confidence: MEDIUM, Severity: HIGH)
    237:         // initialize global pseudo random generator
  > 238:         randToken := tokens[rand.Intn(len(tokens))]
    239:         tokenList = append(tokenList, randToken)

[x/dispensation/test/test_common.go:39] - G404 (CWE-338): Use of weak random number generator (math/rand instead of crypto/rand) (Confidence: MEDIUM, Severity: HIGH)
    38:          index1 := rand.Intn(3-0) + 0
  > 39:          index2 := rand.Intn(3-0) + 0
    40:          var out types.Output

[x/dispensation/test/test_common.go:38] - G404 (CWE-338): Use of weak random number generator (math/rand instead of crypto/rand) (Confidence: MEDIUM, Severity: HIGH)
    37:          address := sdk.AccAddress(crypto.AddressHash([]byte("Output1" + strconv.Itoa(i))))
  > 38:          index1 := rand.Intn(3-0) + 0
    39:          index2 := rand.Intn(3-0) + 0

[x/clp/test/test_common.go:125] - G404 (CWE-338): Use of weak random number generator (math/rand instead of crypto/rand) (Confidence: MEDIUM, Severity: HIGH)
    124:     for i := 0; i < numberOfLp; i++ {
  > 125:         externalToken := tokens[rand.Intn(len(tokens))]
    126:         asset := types.NewAsset(TrimFirstRune(externalToken))

[x/clp/test/test_common.go:112] - G404 (CWE-338): Use of weak random number generator (math/rand instead of crypto/rand) (Confidence: MEDIUM, Severity: HIGH)
    111:         // initialize global pseudo random generator
  > 112:         externalToken := tokens[rand.Intn(len(tokens))]
    113:         externalAsset := types.NewAsset(TrimFirstRune(externalToken))

[x/tokenregistry/utils/parse_denom_list.go:18] - G304 (CWE-22): Potential file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
```

```
27      17:     }
28    > 18:     o, err := ioutil.ReadFile(file)
29      19:     if err != nil {
30
31 [x/ethbridge/client/cli/tx.go:396] - G304 (CWE-22): Potential file
↳  inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
32      395:
33    > 396:            contents, err := ioutil.ReadFile(file)
34      397:            if err != nil {
35
36 [x/dispensation/utils/parser.go:44] - G304 (CWE-22): Potential
↳ file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
37      43:     }
38    > 44:     o, err := ioutil.ReadFile(file)
39      45:     if err != nil {
40
41 [x/dispensation/utils/parser.go:26] - G304 (CWE-22): Potential
↳ file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
42      25:     }
43    > 26:     input, err := ioutil.ReadFile(file)
44      27:     if err != nil {
45
46 [x/clp/test/test_common.go:228] - G304 (CWE-22): Potential file
↳ inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
47      227:    }
48    > 228:    input, err := ioutil.ReadFile(file)
49      229:    if err != nil {
50
51 [x/clp/client/cli/tx.go:432] - G304 (CWE-22): Potential file
↳ inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
52      431:                }
53    > 432:                input, err = ioutil.ReadFile(file)
54      433:                if err != nil {
55
56 [x/clp/client/cli/tx.go:417] - G304 (CWE-22): Potential file
↳ inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
57      416:            }
58    > 417:            input, err := ioutil.ReadFile(file)
59      418:            if err != nil {
60
61 [x/clp/client/cli/tx.go:67] - G304 (CWE-22): Potential file
↳ inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
62      66:             }
63    > 67:             input, err := ioutil.ReadFile(file)
```

```
64      68:             if err != nil {
65
66  [tools/sifgen/network/network.go:388] - G304 (CWE-22): Potential
↳  file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
67      387:        if !validator.Seed {
68    > 388:            input, err := ioutil.ReadFile(srcFile)
69      389:            if err != nil {
70
71  [tools/sifgen/network/network.go:364] - G304 (CWE-22): Potential
↳  file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
72      363:
73    > 364:        file, err := os.Create(configFile)
74      365:        if err != nil {
75
76  [tools/sifgen/network/network.go:355] - G304 (CWE-22): Potential
↳  file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
77      354:
78    > 355:        content, err := ioutil.ReadFile(configFile)
79      356:        if err != nil {
80
81  [tools/sifgen/genesis/genesis.go:134] - G304 (CWE-22): Potential
↳  file inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
82      133:
83    > 134:    body, err := ioutil.ReadFile(genesisPath)
84      135:    if err != nil {
85
86  [scripts/ibc/tokenregistration/main.go:65] - G304 (CWE-22):
↳  Potential file inclusion via variable (Confidence: HIGH, Severity:
↳   MEDIUM)
87      64:             }
88    > 65:             input, err := ioutil.ReadFile(file)
89      66:             if err != nil {
90
91  [cmd/ebrelayer/internal/symbol_translator/symbol_translator.go:28]
↳   - G304 (CWE-22): Potential file inclusion via variable (
↳  Confidence: HIGH, Severity: MEDIUM)
92      27: func NewSymbolTranslatorFromJSONFile(filename string) (*
↳  SymbolTranslator, error) {
93    > 28:     contents, err := ioutil.ReadFile(filename)
94      29:     if err != nil {
95
96  [cmd/ebrelayer/contract/abi.go:41] - G304 (CWE-22): Potential file
↳   inclusion via variable (Confidence: HIGH, Severity: MEDIUM)
97      40:     // Read the file containing the contracts ABI
```

```
98    > 41:      contractRaw, err := ioutil.ReadFile(dir + filePath)
99      42:      if err != nil {
100
101 [tools/sifgen/utils/cli.go:94] - G301 (CWE-276): Expect directory
 ↳ permissions to be 0750 or less (Confidence: HIGH, Severity: MEDIUM
 ↳ )
102     93: func (c CLI) CreateDir(path string) error {
103   > 94:      return os.MkdirAll(path, 0755)
104     95: }
105
106 [x/clp/keeper/migrations.go:78] - G104 (CWE-703): Errors unhandled
 ↳ . (Confidence: HIGH, Severity: LOW)
107     77:          // nolint:errcheck
108   > 78:          m.keeper.SetPool(ctx, &pool)
109     79:      }
110
111 [scripts/ibc/channeldata/main.go:87] - G104 (CWE-703): Errors
 ↳ unhandled. (Confidence: HIGH, Severity: LOW)
112     86:      }
113   > 87:      clientsRes.Body.Close()
114     88:      err = json.Unmarshal(body, &clientResponse)
115
116 [scripts/ibc/channeldata/main.go:35] - G104 (CWE-703): Errors
 ↳ unhandled. (Confidence: HIGH, Severity: LOW)
117     34:      }
118   > 35:      conRes.Body.Close()
119     36:      err = json.Unmarshal(body, &connectionsResponse)
120
121 [scripts/ibc/channeldata/main.go:20] - G104 (CWE-703): Errors
 ↳ unhandled. (Confidence: HIGH, Severity: LOW)
122     19:      }
123   > 20:      res.Body.Close()
124     21:      err = json.Unmarshal(body, &channelsResponse)
125
126 [app/export.go:177] - G104 (CWE-703): Errors unhandled. (
 ↳ Confidence: HIGH, Severity: LOW)
127     176:
128   > 177:     iter.Close()
129     178:
```

Staticcheck - Security Analysis Output Sample:

Listing 13

```
cmd/ebrelayer/relayer/cosmos.go:241:7: should use !bytes.Equal(tx.Data()[0:4], methodID) instead (S1004)
cmd/ebrelayer/relayer/cosmos.go:286:6: should use bytes.Equal(message.CosmosSender, prophecyClaim.CosmosSender) instead (S1004)
cmd/ebrelayer/relayer/ethereum.go:357:4: should use log.Printf(...) instead of log.Println(fmt.Sprintf(...)) (S1038)
cmd/ebrelayer/relayer/ethereum.go:357:16: unnecessary use of fmt.Sprintf (S1039)
x/clp/keeper/msg_server.go:376:2: this value of totalLiquidityFee is never used (SA4006)
x/clp/types/querier.pb.gw.go:16:2: "github.com/golang/protobuf/descriptor" is deprecated: See the "google.golang.org/protobuf/reflect/protoreflect" package for how to obtain an EnumDescriptor or MessageDescriptor in order to programatically interact with the protobuf type system.  (SA1019)
x/clp/types/querier.pb.gw.go:17:2: "github.com/golang/protobuf/proto" is deprecated: Use the "google.golang.org/protobuf/proto" package instead.  (SA1019)
x/clp/types/querier.pb.gw.go:33:9: descriptor.ForMessage is deprecated: Not all concrete message types satisfy the Message interface. Use MessageDescriptorProto instead. If possible, the calling code should be rewritten to use protobuf reflection instead. See package "google.golang.org/protobuf/reflect/protoreflect" for details.  (SA1019)
x/clp/types/types.go:8:2: should use 'return p.ExternalAsset.Validate()' instead of 'if !p.ExternalAsset.Validate() { return false }; return true' (S1008)
x/clp/types/types.go:29:2: should use 'return l.Asset.Validate()' instead of 'if !l.Asset.Validate() { return false }; return true' (S1008)
x/dispensation/types/records.go:60:2: should use 'return len(uc.UserAddress) != 0' instead of 'if len(uc.UserAddress) == 0 { return false }; return true' (S1008)
x/dispensation/types/records.go:71:2: should use 'return ar.DistributionName != ""' instead of 'if ar.DistributionName == "" { return false }; return true' (S1008)
x/tokenregistry/types/query.pb.gw.go:16:2: "github.com/golang/protobuf/descriptor" is deprecated: See the "google.golang.org/protobuf/reflect/protoreflect" package for how to obtain an EnumDescriptor or MessageDescriptor in order to programatically interact with the protobuf type system.  (SA1019)
```

```
14 x/tokenregistry/types/query.pb.gw.go:17:2: "github.com/golang/
↳ protobuf/proto" is deprecated: Use the "google.golang.org/protobuf
↳ /proto" package instead.  (SA1019)
15 x/tokenregistry/types/query.pb.gw.go:33:9: descriptor.ForMessage
↳ is deprecated: Not all concrete message types satisfy the Message
↳ interface. Use MessageDescriptorProto instead. If possible, the
↳ calling code should be rewritten to use protobuf reflection
↳ instead. See package "google.golang.org/protobuf/reflect/
↳ protoreflect" for details.  (SA1019)
```

Errcheck - Security Analysis Output Sample:

Listing 14

```
 1 clp/keeper/calculations_test.go:1311:34:    clpkeeper.
↳ CalculatePoolUnits(
 2 clp/keeper/keeper_test.go:204:47:   clpKeeper.
↳ GetNormalizationFactorFromAsset(ctx, tc.asset)
 3 clp/keeper/migrations.go:78:19: m.keeper.SetPool(ctx, &pool)
 4 dispensation/keeper/distribution.go:48:22:  defer iterator.Close()
 5 ethbridge/keeper/blacklist.go:64:18:    defer iter.Close()
 6 tokenregistry/utils/parser_test.go:32:17:   defer os.Remove(
↳ filepath)
```

THANK YOU FOR CHOOSING

// HALBORN