HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Router Protocol
Date: May 25th, 2022

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Application Code Review and Security Analysis Report for Router Protocol.
Approved By	Evgeniy Bezuglyi | SC Department Head at Hacken OU
Type of Contracts	Cross-chain bridge
Language	Go
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Website	https://www.routerprotocol.com/
Timeline	01.03.2022 - 25.05.2022
Changelog	29.03.2022 - Initial Review 25.05.2022 - Second Review

Table of contents

Introduction

Hacken OÜ (Consultant) was contracted by Router Protocol (Customer) to conduct a Application Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's application.

Scope

The scope of the project is code in the repository:

Repository:
https://github.com/router-protocol/router-bridge/tree/integrate-bsc
Commit:
c29a9a8622a139fa39d33c7847b580434dd7919f
Technical Documentation: Yes, in the repository + whitepaper
Tests: Yes
Source:

```
chains/interfaces.go
chains/substrate/config.go
chains/substrate/types.go
chains/substrate/events.go
chains/substrate/chain.go
chains/substrate/writer.go
chains/substrate/connection.go
chains/substrate/listener.go
chains/ethereum/config.go
chains/ethereum/proposal_data.go
chains/ethereum/events.go
chains/ethereum/chain.go
chains/ethereum/writer.go
chains/ethereum/listener.go
chains/ethereum/writer_methods.go
cmd/router-bridge/account.go
cmd/router-bridge/main.go
config/config.go
config/flags.go
connections/ethereum/connection.go
e2e/substrate/substrate.go
e2e/ethereum/ethereum.go
shared/substrate/methods.go
shared/substrate/query.go
shared/substrate/submit.go
shared/substrate/types.go
shared/substrate/client.go
shared/substrate/events.go
shared/substrate/init.go
shared/logs.go
shared/ethereum/deposit.go
shared/ethereum/deploy.go
shared/ethereum/bridge.go
shared/ethereum/generic.go
shared/ethereum/client.go
shared/ethereum/erc721.go
shared/ethereum/events.go
shared/ethereum/centrifuge.go
shared/ethereum/hash.go
shared/ethereum/erc20.go
```

We have scanned this project for commonly known and more specific vulnerabilities.

Executive Summary

The score measurement details can be found in the corresponding section of the methodology.

Documentation quality

The Customer provided a whitepaper with functional requirements. Technical requirements are incomplete. The total Documentation Quality score is **7** out of **10**.

Code quality

The total CodeQuality score is **6** out of **10**. Some code duplications. Some code is commented out. Tests are outdated and incomplete.

Architecture quality

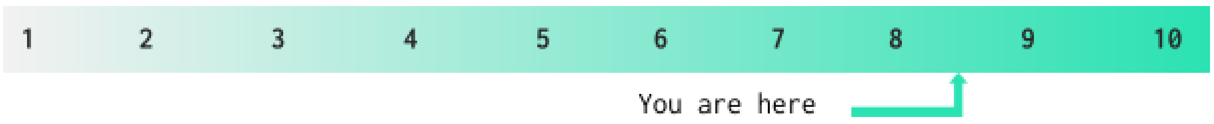
The architecture quality score is **9** out of **10**. All the logic is structured and separated into appropriate files.

Security score

As a result of the audit, security engineers found **1** medium and **2** low severity issues. The security score is **9** out of **10**. All found issues are displayed in the “Issues overview” section.

Summary

According to the assessment, the Customer's project contract has the following score: **8.5**



AS-IS overview

cmd/router-bridge/main.go

Description
Main class for router-bridge application

Imports
- "errors"
- "fmt"
- "math/rand"
- "net/http"
- "os"
- "time"
- "strconv"
- log "github.com/ChainSafe/log15"
- "github.com/prometheus/client_golang/prometheus/promhttp"
- "github.com/router-protocol/router-bridge/chains/ethereum"
- "github.com/router-protocol/router-bridge/chains/substrate"
- "github.com/router-protocol/router-bridge/config"
- "github.com/router-protocol/routerbridge-utils/core"
- "github.com/router-protocol/routerbridge-utils/metrics/health"
- "metrics"
- "github.com/router-protocol/routerbridge-utils/metrics/types"
- "github.com/router-protocol/routerbridge-utils/msg"
- "github.com/spf13/viper"
- "github.com/urfave/cli/v2"

Fields
- app
- cliFlags
- generateFlags
- devFlags
- importFlags
- accountCommand

Structs
main.go has no structs

Functions
- init()
- main()
- startLogger(ctx *cli.Context) error
- run(ctx *cli.Context) error

chains/ethereum/chain.go

Description
Class that setup blockchain connections

Imports
- "fmt"
- "math/big"
- "github.com/router-protocol/routerbridge-utils/blockstore"
- "github.com/router-protocol/routerbridge-utils/core"
- "github.com/router-protocol/routerbridge-utils/crypto/secp256k1"
- "github.com/router-protocol/routerbridge-utils/keystore"
- metrics "github.com/router-protocol/routerbridge-utils/metrics/types"
- "github.com/ChainSafe/log15"
- "github.com/ethereum/go-ethereum/accounts/abi/bind"
- "github.com/ethereum/go-ethereum/common"
- "github.com/ethereum/go-ethereum/ethclient"
- bridge "github.com/router-protocol/router-bridge/bindings/BridgeUpgradeable"
- erc20Handler "github.com/router-protocol/router-bridge/bindings/ERC20HandlerUpgradeable"
- connection "github.com/router-protocol/router-bridge/connections/ethereum"
- "github.com/router-protocol/routerbridge-utils/msg"

Fields
chain.go has no fields

Structs:

```
type Chain struct {
	cfg      *core.ChainConfig // The config of the chain
	conn     Connection        // THe chains connection
	listener *listener         // The listener of this chain
	writer   *writer           // The writer of the chain
	stop     chan<- int
}
```

Functions
- setupBlockstore(cfg *Config, kp *secp256k1.Keypair) (*blockstore.Blockstore, error)
- InitializeChain(chainCfg *core.ChainConfig, logger log15.Logger, sysErr chan<- error, m *metrics.ChainMetrics) (*Chain, error)
- (c *Chain) SetRouter(r *core.Router)

- (c *Chain) Start() error
- (c *Chain) LatestBlock() metrics.LatestBlock

chains/ethereum/config.go

Description:
Encapsulates all necessary parameters in ethereum compatible forms

Imports

- "errors"
- "fmt"
- "math/big"
- "github.com/ethereum/go-ethereum/common"
- utils "github.com/router-protocol/router-bridge/shared/ethereum"
- "github.com/router-protocol/routerbridge-utils/core"
- "github.com/router-protocol/routerbridge-utils/msg"

Fields

- BridgeOpt
- Erc20HandlerOpt
- Erc721HandlerOpt
- GenericHandlerOpt
- MaxGasPriceOpt
- GasLimitOpt
- GasMultiplier
- HttpOpt
- StartBlockOpt
- BlockConfirmationsOpt

Structs:
```
type Config struct {
      name                   string
      id                     msg.ChainId
      endpoint               string
      from                   string
      keystorePath           string
      blockstorePath         string
      freshStart             bool
      bridgeContract         common.Address
      erc20HandlerContract   common.Address
      erc721HandlerContract  common.Address
      genericHandlerContract common.Address
      gasLimit               *big.Int
      maxGasPrice            *big.Int
      gasMultiplier          *big.Float
      http                   bool
      startBlock             *big.Int
      blockConfirmations     *big.Int
}
```

Functions

- parseChainConfig(chainCfg *core.ChainConfig) (*Config, error)

chains/ethereum/events.go

Description
Handle deposit events

Imports

- "github.com/ethereum/go-ethereum/accounts/abi/bind"
- ERC20Handler "github.com/router-protocol/router-bridge/bindings/ERC20HandlerUpgradeable"
- "github.com/router-protocol/routerbridge-utils/msg"

Fields
events.go has no fields

Structs
events.go has no structs

Functions

- (l *listener) handleErc20DepositedEvent(destId msg.ChainId, nonce msg.Nonce) (msg.Message, error)

chains/ethereum/listener.go

Description
Create and setup listener

Imports

- "context"
- "errors"
- "fmt"
- "math/big"
- "strconv"
- "time"
- "github.com/ChainSafe/log15"
- eth "github.com/ethereum/go-ethereum"
- "github.com/ethereum/go-ethereum/accounts/abi/bind"
- ethcommon "github.com/ethereum/go-ethereum/common"
- Bridge "github.com/router-protocol/router-bridge/bindings/BridgeUpgradeable"

- ERC20Handler "github.com/router-protocol/router-bridge/bindings/ERC20HandlerUpgradeable"
- "github.com/router-protocol/router-bridge/chains"
- utils "github.com/router-protocol/router-bridge/shared/ethereum"
- "github.com/router-protocol/routerbridge-utils/blockstore"
- metrics "github.com/router-protocol/routerbridge-utils/metrics/types"
- "github.com/router-protocol/routerbridge-utils/msg"

Fields

- BlockRetryInterval
- BlockRetryLimit
- ErrFatalPolling

Structs

```
type listener struct {
      cfg                   Config
      conn                  Connection
      router                chains.Router
      bridgeContract        *Bridge.BridgeUpgradeable
      erc20HandlerContract *ERC20Handler.ERC20HandlerUpgradeable
      log                 log15.Logger
      blockstore          blockstore.Blockstorer
      stop                <-chan int
      sysErr              chan<- error // Reports fatal error to core
      latestBlock         metrics.LatestBlock
      metrics             *metrics.ChainMetrics
      blockConfirmations *big.Int
}

type DepositData struct {
      chainId uint8
      dataHex [32]byte
      nonce   uint64
}
```

Functions

- NewListener(conn Connection, cfg *Config, log log15.Logger, bs blockstore.Blockstorer, stop <-chan int, sysErr chan<- error, m *metrics.ChainMetrics) *listener
- (l *listener) setContracts(bridge *Bridge.BridgeUpgradeable, erc20Handler*ERC20Handler.ERC20HandlerUpgradeable,)
- (l *listener) setRouter(r chains.Router)
- (l *listener) start() error
- (l *listener) pollBlocks() error
- (l *listener) getDepositEventsForBlock(latestBlock *big.Int) error
- retrieveDepositData(rawdata []ethcommon.Hash) DepositData

- buildQuery(contract ethcommon.Address, sig utils.EventSig, startBlock *big.Int, endBlock *big.Int) eth.FilterQuery

chains/ethereum/proposal_data.go

Description
Construct proposals that pass to blockchain

Imports
- "math/big"
- "github.com/ethereum/go-ethereum/common"
- "github.com/ethereum/go-ethereum/common/math"

Fields
proposal_data.go has no fields

Structs
proposal_data.go has no structs

Functions
- func ConstructErc20ProposalData(srcAmount []byte, stableAmount []byte, destStableAmount []byte, destAmount []byte, recipient common.Address, srcToken []byte, destStableToken []byte, destToken []byte, isDestNative []byte,) []byte
- ConstructErc721ProposalData(tokenId []byte, recipient []byte, metadata []byte) []byte
- ConstructGenericProposalData(metadata []byte) []byte

chains/ethereum/writer.go

Description
Writer to blockchain

Imports
- "github.com/ChainSafe/log15"
- Bridge "github.com/router-protocol/router-bridge/bindings/BridgeUpgradeable"
- "github.com/router-protocol/routerbridge-utils/core"
- metrics "github.com/router-protocol/routerbridge-utils/metrics/types"
- "github.com/router-protocol/routerbridge-utils/msg"

Fields
- PassedStatus
- TransferredStatus
- CancelledStatus

Structs
type writer struct {

```
        cfg            Config
        conn           Connection
        bridgeContract *Bridge.BridgeUpgradeable
        log            log15.Logger
        stop           <-chan int
        sysErr         chan<- error
        metrics        *metrics.ChainMetrics
}
```

Functions

- NewWriter(conn Connection, cfg *Config, log log15.Logger, stop <-chan int, sysErr chan<- error, m *metrics.ChainMetrics) *writer
- (w *writer) start() error
- (w *writer) setContract(bridge *Bridge.BridgeUpgradeable)
- (w *writer) ResolveMessage(m msg.Message) bool

chains/ethereum/writer_methods.go

Description
Contain supplementary functions that interact with blockchain

Imports

- "context"
- "encoding/json"
- "errors"
- "io/ioutil"
- "math/big"
- "math/rand"
- "net/http"
- "os"
- "strconv"
- "strings"
- "time"
- log "github.com/ChainSafe/log15"
- Bridge "github.com/router-protocol/router-bridge/bindings/BridgeUpgradeable"
- ERC20Handler "github.com/router-protocol/router-bridge/bindings/ERC20HandlerUpgradeable"
- "github.com/router-protocol/routerbridge-utils/msg"
- eth "github.com/ethereum/go-ethereum"
- "github.com/ethereum/go-ethereum/accounts/abi"
- "github.com/ethereum/go-ethereum/accounts/abi/bind"
- ethcommon "github.com/ethereum/go-ethereum/common"

- HandlerReserve "github.com/router-protocol/router-bridge/bindings/HandlerReserveUpgradeable"
- utils "github.com/router-protocol/router-bridge/shared/ethereum"

Fields

- ExecuteBlockWatchLimit
- TxRetryInterval
- TxRetryLimit
- ExecuteWatchLimit
- ErrNonceTooLow
- ErrTxUnderpriced
- ErrFatalTx
- ErrFatalQuery

Structs

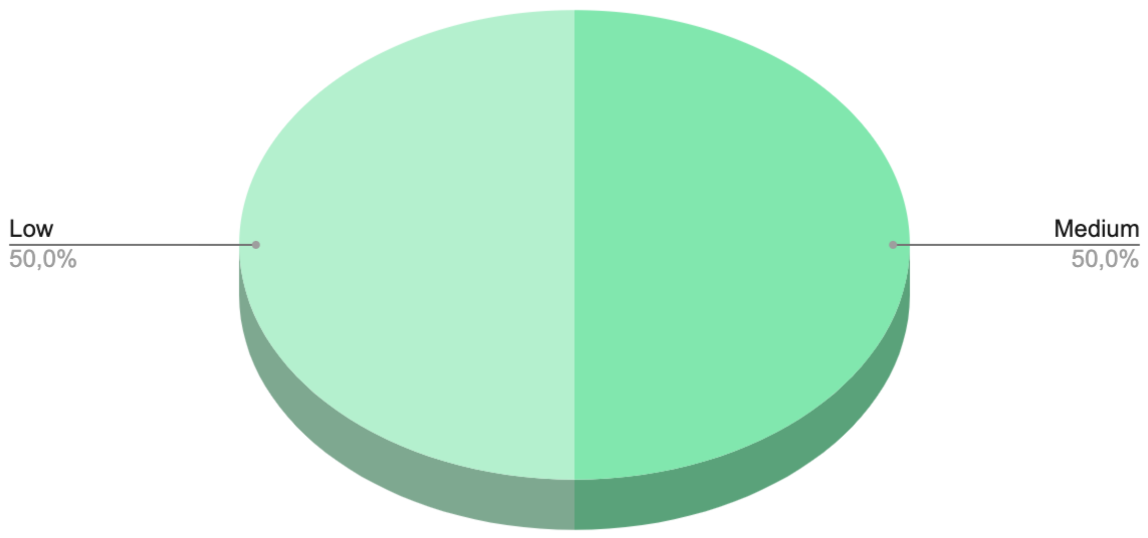
```
type ProposalData struct {
	sourceId     uint8
	depositNonce uint64
	status       uint8
}

type PathFinderAPIResponse struct {
	distribution []*big.Int
	path         []ethcommon.Address
}

type Inner struct {
	TokenAddresses []ethcommon.Address `json:"tokenAddresses"`
}

type Container struct {
	Key Inner `json:"data"`
}
```

Functions

- (w *writer) proposalIsComplete(srcId msg.ChainId, nonce msg.Nonce, dataHash [32]byte) bool
- (w *writer) proposalIsFinalized(srcId msg.ChainId, nonce msg.Nonce, dataHash [32]byte) bool
- (w *writer) proposalIsPassed(srcId msg.ChainId, nonce msg.Nonce, dataHash [32]byte) bool
- (w *writer) hasVoted(srcId msg.ChainId, nonce msg.Nonce, dataHash [32]byte) bool
- (w *writer) shouldVote(m msg.Message, dataHash [32]byte) bool
- (w *writer) createErc20Proposal(m msg.Message) bool

- (w *writer) createErc721Proposal(m msg.Message) bool
- (w *writer) createGenericDepositProposal(m msg.Message) bool
- buildQueryForProposalNew(contract ethcommon.Address, sig utils.EventSig, startBlock *big.Int, endBlock *big.Int) eth.FilterQuery
- retrieveProposalData(rawdata []interface{}) ProposalData
- (w *writer) watchThenExecute(m msg.Message, data []byte, dataHash [32]byte, latestBlock *big.Int)
- (w *writer) voteProposal(m msg.Message, dataHash [32]byte)
- (w *writer) executeProposal(m msg.Message, data []byte, dataHash [32]byte)

Graph 1. The distribution of vulnerabilities after the audit.



Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution

Findings

🟥🟥🟥🟥 Critical

No critical severity issues were found.

🟨🟨🟨 High

1. **Array out of bounds**

In case *m.Destination + chainIndexStart* is not equal to **1**,**2** or **3**, *flags* array will become zero-sized which could cause an error in the contract called in line 637:

```
expectedReturn, err := reserveInstance.GetExpectedReturnWithGasMulti(
  &bind.CallOpts{From: w.conn.Keypair().CommonAddress()},
  oneSplitAddress,
  tokenPath,
  destStableTokenAmount,
  parts,
  flags,
  estimatedGasArr,
)
```

File: chains/ethereum/writer_methods.go

Function: executeProposal

Recommendation: Ensure that all necessary parameters are passed to the smart contract.

Update: as for commit `c29a9a8` the `flags` variable is now **always** zero-sized. Please double-check the function which is being called.

Status: Escalated from Medium to High.

🟩🟩 Medium

1. **Low test coverage and outdated tests**

Some included tests have low test coverage. Others are failing due to incorrect signatures, at least.

It is recommended to cover all functions with tests.

Recommendation: Fix failing tests and implement missing ones. The goal is to raise coverage to a minimum of 95% for branches, while it should be 100% for the main logic. This approach, combined with strict behavior and output control, will help detect most bugs.

Status: Mitigated. Tests are coming in a separate report.

■ Low

1. Different behavior in different environments.

Application behavior is based on *isProduction* variable. This approach could cause errors that could be hard to reproduce.

File: chains/ethereum/writer_methods.go

Recommendation: Do not change execution flow based on environment. If it is necessary to have different sets of chains, put all necessary data in the config file.

Status: Reported.

2. Hardcoded values

There is a lookup table that sets chain id based on network id with hardcoded values. For some values (id = 5) there is different behavior.

File: chains/ethereum/writer_methods.go

Function: executeProposal, proposalIsComplete

Recommendation: Put chain id as a config parameter, and remove hardcoded values.

Status: Reported.

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the audit cannot guarantee the explicit security of the audited smart contracts.

Appendix A. Automatic tools output

Gosec security analysis output:

```
Results:


/prj/router-bridge/chains/ethereum/writer_methods.go:280 - G404 (CWE-338): Use of weak random number
generator (math/rand instead of crypto/rand) (Confidence: MEDIUM, Severity: HIGH)
    279:        w.log.Info("Watching for finalization event", "src", m.Source, "nonce",
m.DepositNonce)
  > 280:        executeWatchLimit := rand.Intn(ExecuteWatchLimit)
    281:        w.log.Info("generate random num: ", "executeWatchLimit", executeWatchLimit)



/prj/router-bridge/config/config.go:53 - G304 (CWE-22): Potential file inclusion via variable
(Confidence: HIGH, Severity: MEDIUM)
    52:
  > 53:          newFile, err = os.Create(file)
    54:          if err != nil {



/prj/router-bridge/cmd/router-bridge/main.go:147 - G104 (CWE-703): Errors unhandled. (Confidence:
HIGH, Severity: LOW)
    146:        os.Setenv("PATHFINDER_API_KEY", fmt.Sprintf("%v", PATHFINDER_API_KEY))
  > 147:        os.Setenv("IS_PRODUCTION", fmt.Sprintf("%v", IS_PRODUCTION))
    148:



/prj/router-bridge/cmd/router-bridge/main.go:146 - G104 (CWE-703): Errors unhandled. (Confidence:
HIGH, Severity: LOW)
    145:        os.Setenv("PATHFINDER_API_URL", fmt.Sprintf("%v", PATHFINDER_API_URL))
  > 146:        os.Setenv("PATHFINDER_API_KEY", fmt.Sprintf("%v", PATHFINDER_API_KEY))
    147:        os.Setenv("IS_PRODUCTION", fmt.Sprintf("%v", IS_PRODUCTION))



/prj/router-bridge/cmd/router-bridge/main.go:145 - G104 (CWE-703): Errors unhandled. (Confidence:
HIGH, Severity: LOW)
    144:
  > 145:        os.Setenv("PATHFINDER_API_URL", fmt.Sprintf("%v", PATHFINDER_API_URL))
    146:        os.Setenv("PATHFINDER_API_KEY", fmt.Sprintf("%v", PATHFINDER_API_KEY))



/prj/router-bridge/cmd/router-bridge/main.go:127 - G104 (CWE-703): Errors unhandled. (Confidence:
HIGH, Severity: LOW)
    126:        viper.SetConfigFile(".env")
  > 127:        viper.ReadInConfig()
    128:        PATHFINDER_API_URL := viper.Get("PATHFINDER_API_URL")



Summary:
  Gosec  : dev
  Files  : 71
  Lines  : 30307
  Nosec  : 0
  Issues : 6
```

Unconvert security analysis output:

```
/prj/router-bridge/chains/ethereum/events.go:31:14: unnecessary conversion
/prj/router-bridge/chains/ethereum/writer_methods.go:263:30: unnecessary conversion
/prj/router-bridge/chains/ethereum/writer_methods.go:265:28: unnecessary conversion
/prj/router-bridge/chains/ethereum/writer_methods.go:268:34: unnecessary conversion
/prj/router-bridge/chains/ethereum/writer_methods.go:319:56: unnecessary conversion
/prj/router-bridge/chains/ethereum/writer_methods.go:337:30: unnecessary conversion
/prj/router-bridge/chains/ethereum/writer_methods.go:345:29: unnecessary conversion
```

Ineffassign security analysis output:

```
/prj/router-bridge/chains/ethereum/events.go:16:12: ineffectual assignment to err
/prj/router-bridge/chains/ethereum/writer_methods.go:53:13: ineffectual assignment to err
/prj/router-bridge/chains/ethereum/writer_methods.go:388:15: ineffectual assignment to err
/prj/router-bridge/chains/ethereum/events.go:16:12: ineffectual assignment to err
/prj/router-bridge/chains/ethereum/writer_methods.go:53:13: ineffectual assignment to err
/prj/router-bridge/chains/ethereum/writer_methods.go:388:15: ineffectual assignment to err
```

All results are reviewed and all detections was sorted out as non-issues.