LUNARAY
BLOCKCHAINSECURITY

CHAIN SECURITY AUDIT REPORT

For Qitmeer

25 November 2021

lunaray.co

Table of Contents

1. Overview......3
2. Background......4
2.1 Project Description......4
2.2 Audit scope......5
2.3 Findings Summary......7
3. Project Contract Details......8
3.1 Directory Structure......8
3.2 Ledger structure......13
3.3 RPC interface list......15
3.4 P2P protocol list......16
3.5 External reference library......17
4. Audit Details......18
4.1 Risk Distribution......18
4.2 Risk Audit Details......19
4.2.1 P2P......19
4.2.2 RPC interface related......20
4.2.3 Static audit......26
5. Security Audit Tool......27

1. Overview

The Lunaray security team conducted a security audit on the Qitmeer public chain project in November 2021. This audit mainly included static code security and analysis of the static code content and the security of reference libraries, dependent libraries, and RPC. In this audit, there were no serious security issues in the code itself, and there were some specification issues in code specifications. Static code and memory allocation each had a security risk.

The Qitmeer public chain audit result: **Passed**

Audit Report MD5：059CBC184386A0FCEE50703B5B2601A8

2. Background

2.1 Project Description

Project name	Qitmeer
Project address	https://qitmeer.io/
Code warehouse	https://github.com/Qitmeer/qitmeer
Audit version	commit f1fa7ede24ea722d88f29bf612eda347f9ea1108
Code language	Golang
Project Description	Qitmeer is the next generation public chain based on BlockDAG which is dedicated to serving the ecosystem of Islamic Finance, ethical finance, and socially responsible investment, thereby enhancing financial inclusion and creating social impact.

2.2 Audit scope

The list of Qitmeer public chain audit projects is as follows：

Type	Name
Environment setup	Main chain construction and debugging
Code audit	Golang static code audit
RPC	RPC interface test
P2P	P2P protocol test
Safety of tradement	Fake recharge attack
Private key	Private key security
P2P	Sybil Attack
P2P	Denial of service test
P2P	Large handshake package test
P2P	Slow attack
P2P	Eclipse attack
P2P	Multi-connection test
P2P	Fuzz test

RPC	Cross-domain resource sharing (CORS) testing
RPC	Interface certification test
RPC	Secure transmission test
RPC	Super deep JSON attack
RPC	Oversized JSON Key attack
RPC	Oversized JSON Value attack
RPC	Fuzz test
Consensus security	Block verification
Consensus security	Transaction verification
Consensus security	Transaction replay attack
Merkle Tree	Replay attack
Merkle Tree	Transaction malleability attack

2.3 Findings Summary

Severity	Found	Resolved	Acknowledged
● High	0	0	0
● Medium	0	0	0
● Low	2	2	0
● Info	0	0	0

3. Project Contract Details

3.1 Directory Structure

```
├──common
| ├──encode
| | ├──base58
| | ├──bech32
| | ├──leb128
| | └──rlp
| ├──hash
| |  ├──btc
| |  └──dcr
| ├──marshal
| ├──math
| ├──network
| ├──prque
| └──util
├──config
├──consensus
├──container
|  └──docker
|        └──alpine
├──core
|  ├──address
|   ├──blockchain
|   ├──blockdag
|   |   └──anticone
|   ├──dbnamespace
|   ├──json
|   ├──merkle
|   ├──message
|   ├──protocol
|   ├──serialization
|   └──types
|         └──pow
├──crypto
|   ├──bip32
|   ├──bip39
|   |   └──wordlists
|   ├──certgen
|   ├──cuckoo
|   |   └──siphash
|   ├──ecc
|   |   ├──ed25519
|   |   |   ├──internal
|   |   |   |   ├──edwards25519
|   |   |   |   └──testdata
|   |   |   └──testdata
|   |   ├──schnorr
|   |   └──secp256k1
|   └──seed
├──database
|   ├──benchmark
|   ├──ffldb
|   |   └──treap
|   └──statedb
├──engine
|   └──txscript
├──ledger
├──log
|   └──term
├──metrics
├──node
|   └──notify
├──p2p
|   ├──addmgr
|   ├──connmgr
|   ├──peer
|   |   ├──invcache
|   |   └──nounce
|   └──peerserver
├──params
```

```
├──qx
├──rpc
├──script
├──services
│   ├──acct
│   ├──blkmgr
│   ├──bloom
│   ├──cf
│   ├──common
│   │   └──progresslog
│   ├──index
│   ├──mempool
│   ├──miner
│   ├──mining
│   ├──notifymgr
│   └──tx
├──tools
│   ├──findcheckpoint
│   ├──ngen
│   ├──nx-eth
│   ├──nx-hd
│   ├──payledger
│   ├──qx
│   │   └──bash_completion
```

```
|   └──rlpdump
├──trie
├──version
└──wallet
```

3.2 Ledger structure

```
// Transaction
type Transaction struct
{
    Version uint32
    TxIn    []*TxInput
    TxOut   []*TxOutput
    LockTime uint32
    Expire  uint32
    Message     []byte
    CachedHash *hash.Hash
}

// Contract transaction
type ContractTransaction struct
{
    From Account
    To Account
    Value uint64
    GasPrice uint64
    GasLimit uint64
    Nonce uint64
    Input []byte
    Signature []byte
}

// Block node
type blockNode struct
{
    parents
    []*blockNode children
    []*blockNode hash
    hash.Hash workSum
    *big.Int
    blockVersion uint32
    bits    uint32
    timestamp   int64
    txRoot  hash.Hash
    stateRoot   hash.Hash
    extraData   [32]byte
    status blockStatus
    order uint64
    height uint
    layer uint
    pow pow.IPow
    dirty bool
}

// Block header
type BlockHeader struct
{
     Version uint32
     ParentRoot hash.Hash
     TxRoot hash.Hash
     StateRoot hash.Hash
     Difficulty uint32
     Timestamp time.Time
     Pow pow.IPow
}
```

3.3 RPC interface list

getBlockCount

getBlockHash

getBlock

getBlockHashByRange

getBlockByOrder

getBestBlockHash

getBlockHeader

isOnMainChain

getMainChainHeight

getBlockWeight

createRawTransaction

getRawTransaction

decodeRawTransaction

sendRawTransaction

txSign

getUtxo

getNodeInfo

getPeerInfo

getMempool

generate

getBlockTemplate

submitBlock

3.4 P2P protocol list

version

verack

getaddr

addr

reject

ping

pong

inv

block

getblocks

headers

miningstate

mempool

graphstate

sendheaders

feefilter

getcfilter

getcfheaders

getcftypes

cfilter

cfheaders

cftypes

3.5 External reference library

github.com/davecgh/go-spew v1.1.1

github.com/dchest/blake256 v1.0.0

github.com/deckarep/golang-set v1.7.1 github.com/go-stack/stack v1.8.0

github.com/golang-collections/collections v0.0.0-20130729185459-604e922904d3

github.com/jessevdk/go-flags v1.4.0

github.com/jrick/logrotate v1.0.0

github.com/mattn/go-colorable v0.1.1 github.com/pkg/errors v0.8.1

github.com/rcrowley/go-metrics v0.0.0-20181016184325-3113b8401b8a

github.com/satori/go.uuid v1.2.0

github.com/stretchr/testify v1.3.0 github.com/syndtr/goleveldb v1.0.0

golang.org/x/crypto v0.0.0-20190621222207-cc06ce4a13d4

golang.org/x/net v0.0.0-20190503192946-f4e77d36d62c

golang.org/x/sys v0.0.0-20190412213103-97732733099d

golang.org/x/tools v0.0.0-20190511041617-99f201b6807e

gonum.org/v1/gonum v0.0.0-20190608115022-c5f01565d866

4. Audit Details

4.1 Risk Distribution

Name	Risk level	Status
Memory allocation	Low	Resolved
Interface certification test	No	Passed
Secure transmission test	No	Passed
Super deep JSON attack	No	Passed
Oversized JSON Key attack	No	Passed
Oversized JSON Value attack	No	Passed
RPC cross-domain (CORS) testing	No	Passed
Zero division risk	Low	Resolved

4.2 Risk Audit Details

4.2.1 P2P

4.2.1.1 Memory allocation

- **Risk description**

During the audit, it was found that most of the memory allocation size is constant or uncontrollable, but there are some methods that use make for memory allocation without restricting the size. It is recommended to limit the size during memory allocation to avoid unpredictable risks.

```
func CheckEncode(input []byte, version []byte, cksum_size int, cksumfuncfunc([]byte) []byte) string {

        b := make([]byte, 0, len(version)+len(input)+cksum_size)b = append(b,
        version[:]...)

        b = append(b, input[:]...)

        var cksum []byte = cksumfunc(b)b =
        append(b, cksum[:]...) return Encode(b)
```

- **Safety advice**

It is recommended to limit the size during memory allocation to avoid unpredictable risks.

- **Repair status**

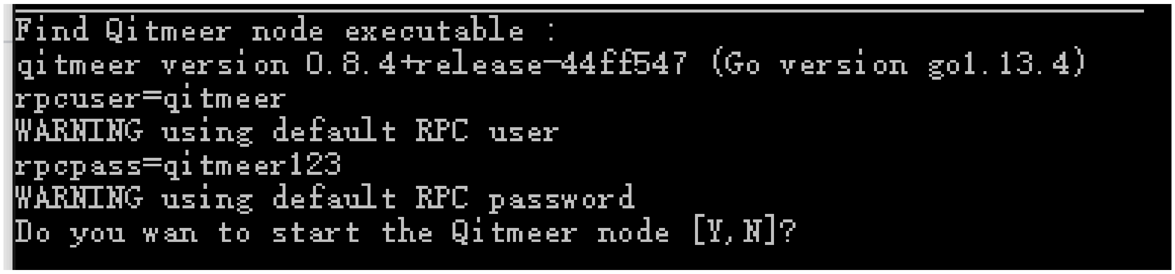
Qitmeer official has confirmed to fix the risk.

4.2.2 RPC interface related

4.2.2.1 Interface certification test

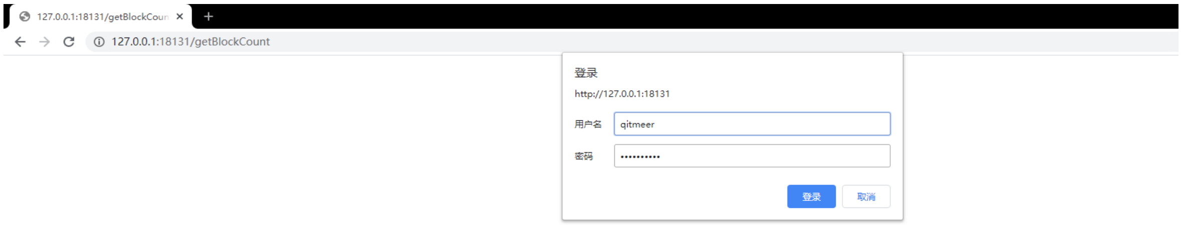
- **Risk description**

When starting the service, if the RPC default password is not used, a warning will be issued.



```
Find Qitmeer node executable :
qitmeer version 0.8.4+release-44ff547 (Go version go1.13.4)
rpcuser=qitmeer
WARNING using default RPC user
rpcpass=qitmeer123
WARNING using default RPC password
Do you wan to start the Qitmeer node [Y,N]?
```

Login authentication is required when calling the RPC interface.



127.0.0.1:18131/getBlockCoun
127.0.0.1:18131/getBlockCount

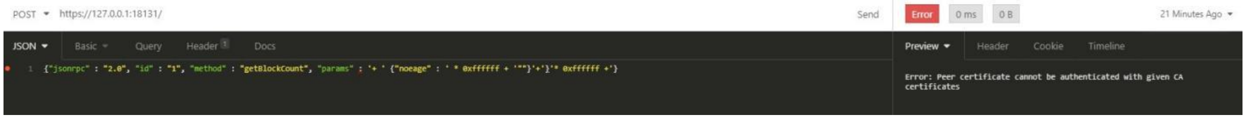
登录
http://127.0.0.1:18131
用户名 qitmeer
密码 ••••••••••

- **Audit Results : Passed**

4.2.2.2 Secure transmission test

- **Risk description**

HTTPS protocol transmission is adopted in the default configuration to prevent man-in-the-middle attacks.



https://127.0.0.1:18131/

```
{"jsonrpc" : "2.0", "id" : "1", "method" : "getBlockCount", "params" : '+ ' {"noeage" : ' * 0xffffff + '""}'+'}'* 0xffffff +'}
```

```
Error: Peer certificate cannot be authenticated with given CA
certificates
```

- **Audit Results : Passed**

4.2.2.3 Super deep JSON attack

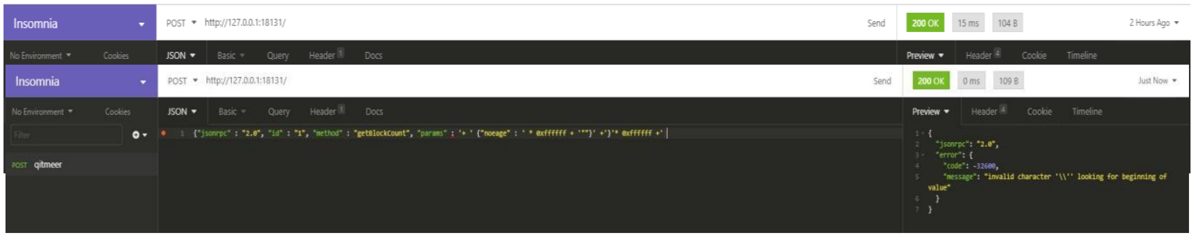
- **Risk description**

Ultra-large deep JSON attack, constructing malformed data packets will not cause the interface to feign death.

```
{"jsonrpc" : "2.0", "id" : "1", "method" : "getBlockCount", "params" : '+ '

{"noeage" : ' * 0xffffff + '""}'+'}'* 0xffffff +'}
```

- **Audit Results : Passed**



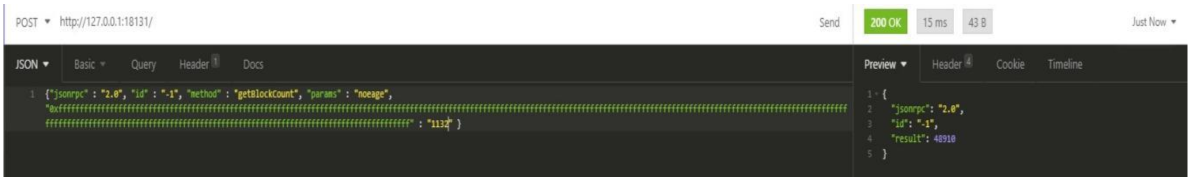
```
{"jsonrpc" : "2.0", "id" : "1", "method" : "getBlockCount", "params" : '+ ' {"noeage" : ' * 0xffffff + '""}' +'}'* 0xffffff +'
```

```
{
  "jsonrpc": "2.0",
  "error": {
    "code": -32600,
    "message": "invalid character '\\'' looking for beginning of value"
  }
}
```

4.2.2.4 Oversized JSON Key attack

- **Risk description**

Oversized JSON Key attack, constructing oversized JSON Key data packets will not cause the interface to feign death。



```
{"jsonrpc" : "2.0", "id" : "-1", "method" : "getBlockCount", "params" : "noeage",
"0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff" : "1132" }
```

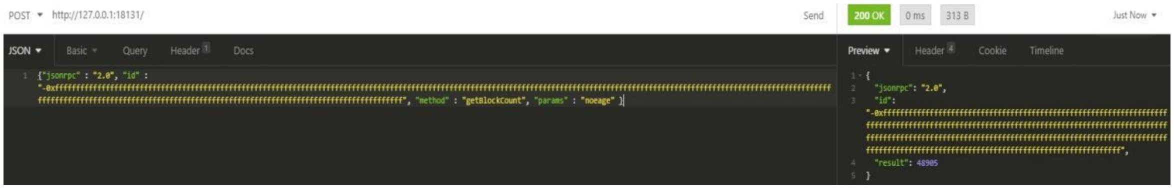
```
{
  "jsonrpc": "2.0",
  "id": "-1",
  "result": 48910
}
```

- **Audit Results : Passed**

4.2.2.5 Oversized JSON Value attack

- **Risk description**

Oversized JSON Value attacks, constructing oversized JSON Value data packets will not cause the interface to feign death.



```
http://127.0.0.1:18131/
```

```
{"jsonrpc" : "2.0", "id" :
"-0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff", "method" : "getBlockCount", "params" : "noeage" }
```

```
{
  "jsonrpc": "2.0",
  "id":
"-0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffff",
  "result": 48905
}
```

- **Audit Results : Passed**

4.2.2.6 RPC cross-domain (CORS) testing

- **Risk description**

When qitmeer calls the RPC service, login authentication is required by default, and HTTPS is used by default.

```
var xhr = new XMLHttpRequest(); xhr.open("POST",
"http://127.0.0.1",true);

xhr.setRequestHeader("Content-Type", "application/json"); xhr.setRequestHeader('Authorization', 'Basic
cWl0bWVlcjpxaXRtZWVyMTIz');xhr.onreadystatechange = function() {

      if (xhr.readyState == XMLHttpRequest.DONE && xhr.status == 200) {console.log("Modules:
      " + xhr.responseText);

      }
```

qitmeer related RPC interface, with a certain method of processing illegal data, and verify the identity when calling.

- **Audit Results : Passed**

4.2.3 Static audit

4.2.3.1 Zero division risk

- **Risk description**

In Go, dividing by zero results in a panic. During the audit, it was found that most of the denominators are const and constant, but some methods have variables as the denominator. Whether there is a division by zero depends on the caller:

```
func GetRisk(N int, alpha float64, lambda float64, delay float64, waitingTimeuint, antiPast int) float64 {

        ...

vecData:=[]float64{}var
        vecMod float64

        for i := 0; i < r; i++ {

                realData:=real(ceigenvectors.At(i,featuresIndex))vecData =
                append(vecData, realData)

                vecMod += realData

        }

        vecRMod := 1 / vecMod

        vect:=mat.NewVecDense(r,vecData)
        vect.ScaleVec(vecRMod, vect)
```

There is no use method that can cause the node to crash, but it is not safe to rely on the caller to ensure that it is not zero.

- **Safety advice**

It is recommended that all non-constant dividends should be checked before division.

- **Repair status**

Qitmeer official has confirmed to fix the risk.

5. Security Audit Tool

Tool name	Tool Features
Lunaray Internal Security Toolkit	Lunaray (Eagle Eye System) self-developed security audit toolkit
Lunaray code automation security audit tool	Support code security audit for C/C++, go, java, python, solidity languages

Disclaimer：

Lunaray only issues a report and assumes corresponding responsibilities for the facts that occurred or existed before the issuance of this report. As the safety status of the project cannot be judged for the facts after the issuance of the report, it is not responsible for this. The subsequent on-chain deployment and operation methods of the project party are outside the scope of this audit. This report is only based on the security audit of the information provided by the information provider to Lunaray at the time the report is issued. If the information of this project is concealed or the situation reflected does not match the actual situation, Lunaray will not be responsible for the losses and adverse effects caused thereby. Take any responsibility.

There are risks in the market, and investment needs to be cautious. This report only conducts security audits and results announcements on the project code, and does not make investment recommendations and basis.

LUNARAY
BLOCKCHAINSECURITY

https://lunaray.co

https://github.com/lunaraySec

https://twitter.com/lunaray_Sec

http://t.me/lunaraySec