

SALUS SECURITY

SEP 2024



# CODE SECURITY ASSESSMENT

LORENZO PROTOCOL

# Overview

## Project Summary

- Name: Lorenzo Protocol - lorenzo
- Language: Go
- Repository:
  - <https://github.com/Lorenzo-Protocol/lorenzo>
- Audit Range: See [Appendix - 1](#)

## Project Dashboard

### Application Summary

Name	Lorenzo Protocol - lorenzo
Version	v1
Type	Go
Dates	Sep 24 2024
Logs	Jul 05 2024; Sep 24 2024

### Vulnerability Summary

Total High-Severity issues	2
Total Medium-Severity issues	6
Total Low-Severity issues	0
Total informational issues	0
Total	8

## Contact

E-mail: [support@salusec.io](mailto:support@salusec.io)

## Risk Level Description

<b>High Risk</b>	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for clients' reputations or serious financial implications for clients and users.
<b>Medium Risk</b>	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to a moderate financial impact.
<b>Low Risk</b>	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
<b>Informational</b>	The issue does not pose an immediate risk, but is relevant to security best practices or defense in depth.

# Content

<b>Introduction</b>	<b>4</b>
1.1 About SALUS	4
1.2 Audit Breakdown	4
1.3 Disclaimer	4
<b>Findings</b>	<b>5</b>
2.1 Summary of Findings	5
2.2 Notable Findings	6
1. Lorenzo could possibly incur a loss of funds since it accepts dust amounts outputs	6
2. The HTTP/2 protocol in Golang prior to 1.21.9 is vulnerable to DoS	8
3. Some users might get their stBTC stuck due to the incomplete handling of OP_RETURN pushed data	9
4. The ibc-go v7.0.0 contains a security vulnerability impacting IBC connected full nodes	11
5. Vulnerable http2 library dependency used in grpc v1.61.0	12
6. Potential loss of stBTC for the user in case the receivers are updated before stBTC is minted	13
7. The gasMeter in runTx is not zero before TX execution starts	14
8. CometBFT v0.37.2 contains security vulnerabilities such as rolling back at a specific height could the node to fail on start	15
2.3 Informational Findings	16
<b>Appendix</b>	<b>17</b>
Appendix 1 - Files in Scope	17

# Introduction

## 1.1 About SALUS

At Salus Security, we are in the business of trust.

We are dedicated to tackling the toughest security challenges facing the industry today. By building foundational trust in technology and infrastructure through security, we help clients to lead their respective industries and unlock their full Web3 potential.

Our team of security experts employ industry-leading proof-of-concept (PoC) methodology for demonstrating smart contract vulnerabilities, coupled with advanced red teaming capabilities and a stereoscopic vulnerability detection service, to deliver comprehensive security assessments that allow clients to stay ahead of the curve.

In addition to smart contract audits and red teaming, our Rapid Detection Service for smart contracts aims to make security accessible to all. This high calibre, yet cost-efficient, security tool has been designed to support a wide range of business needs including investment due diligence, security and code quality assessments, and code optimisation.

We are reachable on Telegram (<https://t.me/salusec>), Twitter ([https://twitter.com/salus\\_sec](https://twitter.com/salus_sec)), or Email ([support@salusec.io](mailto:support@salusec.io)).

## 1.2 Audit Breakdown

The objective was to evaluate the repository for security-related issues, code quality, and adherence to specifications and best practices. Possible issues we looked for included (but are not limited to):

- Risky external calls
- Integer overflow/underflow
- Transaction-ordering dependence
- Timestamp dependence
- Access control
- Call stack limits and mishandled exceptions
- Number rounding errors
- Centralization of power
- Logical oversights and denial of service
- Business logic specification
- Code clones, functionality duplication

## 1.3 Disclaimer

Note that this security audit is not designed to replace functional tests required before any software release and does not give any warranties on finding all possible security issues with the given smart contract(s) or blockchain software, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues.

# Findings

## 2.1 Summary of Findings

ID	Title	Severity	Category	Status
1	Lorenzo could possibly incur a loss of funds since it accepts dust amounts outputs	High	Business Logic	Resolved
2	The HTTP/2 protocol in Golang prior to 1.21.9 is vulnerable to DoS	High	Configuration	Resolved
3	Some users might get their stBTC stuck due to the incomplete handling of OP_RETURN pushed data	Medium	Business Logic	Resolved
4	The ibc-go v7.0.0 contains a security vulnerability impacting IBC connected full nodes	Medium	Configuration	Resolved
5	Vulnerable http2 library dependency used in grpc v1.61.0	Medium	Configuration	Resolved
6	Potential loss of stBTC for the user in case the receivers are updated before stBTC is minted	Medium	Business Logic	Resolved
7	The gasMeter in runTx is not zero before TX execution starts	Medium	Business Logic	Resolved
8	CometBFT v0.37.2 contains security vulnerabilities such as rolling back at a specific height could the node to fail on start	Medium	Configuration	Resolved

## 2.2 Notable Findings

Significant flaws that impact system confidentiality, integrity, or availability are listed below.

### 1. Lorenzo could possibly incur a loss of funds since it accepts dust amounts outputs

Severity: High

Category: Business Logic

Target:

- btcstaking/keeper/msg\_server.go

### Description

The user sends BTC to a certain address controlled by Lorenzo. Then, makes a staking call to Lorenzo chain providing the proof. After verifying the proof, it goes through the transaction outputs and calculate the sum of the corresponding ones.

btcstaking/keeper/msg\_server.go:L162-L165

```
if err != nil || btcAmount == 0 {  
    return nil, types.ErrInvalidTransaction  
}
```

If the sum is zero, return invalid TX.

However, if you check extract payment functions `extractPaymentToWithOpReturnId` and `extractPaymentTo`, it doesn't exclude the values with dust amounts. This could cause a loss for Lorenzo as the cost of spending those amounts will be higher than the actual received amounts.

A malicious party can craft a TX with dust amounts to mint stBTC. Thus, Lorenzo will incur a loss of funds since spending those dust amounts is economically unprofitable and worthless, while the malicious party received already the stBTC which can be redeemed in future versions of the protocol.

While transactions with dust amounts are deemed to be non-standard, therefore, are not relayed by Bitcoin Core. Non-standard transactions can still be mined by other mining pools. Please note, an output is considered dust if the pay to spend it is more than the value of the output.

<https://github.com/bitcoin/bitcoin/blob/080a4.../src/policy/policy.cpp#L28-L31>

```
// "Dust" is defined in terms of dustRelayFee,  
// which has units satoshis-per-kilobyte.  
// If you'd pay more in fees than the value of the output  
// to spend something, then we consider it dust.
```

### Recommendation

Don't take dust amounts into account when summing up TX outputs values. Here is a reference from [Bitcoin Core](#).

## **Status**

The team has resolved this issue in commit [d073df8](#).



## 2. The HTTP/2 protocol in Golang prior to 1.21.9 is vulnerable to DoS

Severity: High

Category: Configuration

Target:

- go.mod

### Description

A vulnerability was discovered with the implementation of the HTTP/2 protocol in Golang prior to 1.21.9. The current version used is [go 1.20](#)

An attacker can cause the HTTP/2 endpoint to read arbitrary amounts of header data by sending unlimited number of CONTINUATION frames. This cause excessive CPU consumption of the node since there is no sufficient limitation on the amount of frames. Therefore, if this is exploited, it could lead to DoS.

### Recommendation

Upgrade to a newer version. For instance, 1.21.12 which has fixes of other security issues as well.

### Status

The team has resolved this issue in commit [b7ddbc9](#).

### 3. Some users might get their stBTC stuck due to the incomplete handling of OP\_RETURN pushed data

Severity: Medium

Category: Business Logic

Target:

- btcstaking/keeper/msg\_server.go

## Description

On staking, the function `extractPaymentToWithOpReturnId` is responsible of extracting two inputs from a BTC TX:

- 1- The total amount of Satoshi sent.
- 2- The address to mint the stBTC to.

To extract the address, the data is attached after OP\_RETURN (0x6a) as follows:

```
OP_RETURN <optional data>
```

To push the data, any of the following opcodes can be used:

```
OP_DATA_1
OP_DATA_2
.
.
.
OP_DATA_75
OP_PUSHDAT1
OP_PUSHDAT2
OP_PUSHDAT4
.
.
OP_TRUE/OP_1
OP_2
.
.
OP_16
```

Here are some examples to push an address of 20 bytes:

- With OP\_DATA\_XX  
`OP_RETURN OP_DATA_20 ADDRESS_HERE`
- With OP\_PUSHDAT1  
`OP_RETURN OP_PUSHDAT1 14 ADDRESS_HERE`
- With OP\_PUSHDAT2  
`OP_RETURN OP_PUSHDAT2 0014 ADDRESS_HERE`

However, looking at the function `extractPaymentToWithOpReturnId`, it doesn't handle the case of opcodes OP\_PUSHDAT2 or OP\_PUSHDAT4.

`btcstaking/keeper/msg_server.go:L86-L93`

```
// if this is OP_PUSHDAT1, we need to drop first 3 bytes as those are related
```

```
// to script itself i.e OP_RETURN + OP_PUSHDATA1 + len of bytes
if pkScript[1] == txscript.OP_PUSHDATA1 {
    opReturnId = pkScript[3:]
} else {
    // this should be one of OP_DATAXX opcodes we drop first 2 bytes
    opReturnId = pkScript[2:]
}
```

As a result, some protocols or users will get their stBTC stuck since the address won't be discovered, therefore, stBTC won't be minted and the transaction will be deemed to be invalid.

btcsaking/keeper/msg\_server.go:L162-L164

```
if err != nil || btcAmount == 0 {
    return nil, types.ErrInvalidTransaction
}
```

## Recommendation

Apply the necessary changes to handle OP\_PUSHDATA2 and OP\_PUSHDATA4 opcodes. Note: OP\_TRUE/OP\_1 and OP\_2 through OP\_16 is unnecessary to handle, since the max value is 16 which can't represent a valid length of address.

## Status

The team has resolved this issue in commit [6230e6](#).

#### 4. The ibc-go v7.0.0 contains a security vulnerability impacting IBC connected full nodes

Severity: Medium

Category: Configuration

Target:

- go.mod

### Description

The ibc-go v7.0.0 used by Lorenzo, contains a [security vulnerability](#) impacting IBC connected full nodes.

go.mod:L14

```
github.com/cosmos/ibc-go/v7 v7.0.0
```

### Recommendation

Update ibc-go to a higher version.

Note: please check compatibility with Cosmos SDK versions when doing so

### Status

The team has resolved this issue in commit [b7ddbc9](#).

## 5. Vulnerable http2 library dependency used in grpc v1.61.0

Severity: Medium

Category: Configuration

Target:

- go.mod

### Description

The grpc [v1.61.0](#) has an outdated http2 library dependency. This library has a vulnerability that can be utilized for DoS attacks. Check this for more info:

<https://www.kb.cert.org/vuls/id/421644>

### Recommendation

Update grpc to 1.62.2. Please check for compatibility with the client app in case there is any used.

### Status

The team has resolved this issue in commit [b7ddbc](#).

## 6. Potential loss of stBTC for the user in case the receivers are updated before stBTC is minted

Severity: Medium

Category: Business Logic

Target:

- btcstaking/keeper/msg\_server.go

### Description

When the user makes a staking call to Lorenzo chain, the user should provide the address where the BTC sent to. In Lorenzo, this is called the Receiver.

If the Receiver has EthAddr, then the stBTC is minted to it. Otherwise, Lorenzo search for OP\_RETURN to extract the address from it.

As a conclusion, if the user wants to mint stBTC to an address provided in OP\_RETURN data, the user has to send BTC to a Lorenzo receiver that doesn't have valid Ethereum address.

btcstaking/keeper/msg\_server.go:L156-L162

```
if common.IsHexAddress(receiver.EthAddr) {  
    mintToAddr = common.HexToAddress(receiver.EthAddr).Bytes()  
    btcAmount, err = extractPaymentTo(stakingMsgTx, btc_receiving_addr)  
} else {  
    btcAmount, mintToAddr, err = extractPaymentToWithOpReturnId(stakingMsgTx,  
    btc_receiving_addr)  
}
```

However, if the privileged role which can update receivers, removed a receiver and added it again with a valid Ethereum address before the user makes a call to mint stBTC, the user will lose their stBTC as they will be minted to an address belongs to Lorenzo.

### Recommendation

Add a locktime for updating receivers. This way, the user is protected from such cases. Another suggestion, is to prevent removing receivers, instead, enabling it or disabling it.

### Status

The team has resolved this issue in commit [ed6adde](#).

## 7. The gasMeter in runTx is not zero before TX execution starts

Severity: Medium

Category: Business Logic

Target:

- go.mod

### Description

The blockGasMeter tracks all gas consumed across block execution (excluding BeginBlock and EndBlock), while the gasMeter tracks gas consumed in TX execution. However, the gasMeter in runTx is not zero before TX execution starts. This [bug](#) exists in cosmos-sdk versions prior to v0.47.7. Lorenzo chain utilizes cosmos-sdk [v0.47.4](#). Thus, it has this bug.

### Recommendation

The GasMeter should be zero when processing the first TX in each block. Upgrading to cosmos-sdk v0.47.7 or higher should fix the issue.

### Status

The team has resolved this issue in commit [b7ddbc9](#).

## 8. CometBFT v0.37.2 contains security vulnerabilities such as rolling back at a specific height could the node to fail on start

Severity: Medium

Category: Configuration

Target:

- go.mod

### Description

Lorenzo chain utilizes CometBFT v0.37.2

go.mod:L14

<https://github.com/cometbft/cometbft%20v0.37.2>

However, this version contains security vulnerabilities such as:

The calculation method of tx size returned by calling proxyapp should be consistent with that of mempool ([fixed in v0.37.5](#), fix: Tx Validate #1687)

Fix rollback to a specific height (fixed in v0.37.6 fix: [Rollback #2136](#))

### Recommendation

Update CometBFT to v0.37.6. This shouldn't introduce any compatibility issues according to CometBFT docs.

Patch releases of the same minor release series (v1.0.1, v1.0.2, etc.) are guaranteed to be compatible with each other.

### Status

The team has resolved this issue in commit [b7ddbc9](#).



## 2.3 Informational Findings

No informational issues were found.

# Appendix

## Appendix 1 - Files in Scope

This audit covered the following files in commit [93642bc](#):

File	SHA-1 hash
x/btclient/client/cli/query.go	24524f66eb1f54f1646d85d15c13bcb0ced32109
x/btclient/client/cli/tx.go	5cdbff1b399b8669123e47cbdc62c5c879455a8d
x/btclient/keeper/base_btc_header.go	b8a073e705629a92bb791acb4c7a2f2d515cf773
x/btclient/keeper/grpc_query.go	41c697d072b3e8f240993d3c403fbd2025c47952
x/btclient/keeper/hooks.go	c8ba909596619fab30ebaee0c6a91c2bfea2d515
x/btclient/keeper/keeper.go	ca6b31bbca6d855d7bed916cd0c6fac1c00fb29b
x/btclient/keeper/msg_server.go	ef68668294b940de03499f613440359091bef27b
x/btclient/keeper/params.go	efa5e42278729fc886fcfb2fd60794f3bfb5831f
x/btclient/keeper/state.go	9615fda4a5b1292618cf6c46ed55f50e3bdbd349
x/btclient/keeper/triggers.go	02e4794a67232527840a5b5979a3e4826d8e2df8
x/btclient/keeper/utils.go	fbf891e0a22228c523c9340bea53b6d0e3601ddf
x/btclient/types/btc_header_info.go	c4a23a63da460de9c7c12f7312e9127673c0eb6b
x/btclient/types/btc_light_client.go	8ef1558bde803ef3d90049115bddca66b59c8192
x/btclient/types/btclient.pb.go	5ba313c4da708183ff6e4e6d8c64d620fd9ee344
x/btclient/types/codec.go	5693b9c4e036b6c073572f1e50e0d2ee6ab5a36e
x/btclient/types/errors.go	7405d7b2f15a4603a0276607c254eac0e9696886
x/btclient/types/event.pb.go	a917215338a7c30c52911254677e9bd66ca63b6d
x/btclient/types/expected_keepers.go	ee555f13d9dae473643d7e3012de46c7e00f0e55
x/btclient/types/genesis.go	2e5bca3bf497126b07900573bb831b8a388c6f93
x/btclient/types/genesis.pb.go	a9797251bd0dc3db7bc1b92c9111ed86b37ff700
x/btclient/types/hooks.go	609fb2169768cd06108619147f28c22e4380fab6
x/btclient/types/keys.go	049b00e228f96abc99824cfa187b6d9083a75539
x/btclient/types/msgs.go	300d1f2e4a40a6302845720f61379083f7e5d475

x/btclightclient/types/params.go	28d097dd2290b3ed90c6b2746b83582c26d7b860
x/btclightclient/types/params.pb.go	72700280bfabe9dde8d6a58bb7ac5cd93da056d9
x/btclightclient/types/querier.go	68df427f0350a1040dbc3269b2719894f0c1fd5c
x/btclightclient/types/query.pb.go	902625104a3c3cf46ac6f020172198f91b9918f8
x/btclightclient/types/query.pb.gw.go	0e3bafd37a2717dcfbcd34ac8fb0aa9c4344f91
x/btclightclient/types/tx.pb.go	6bfe8d702ae829c8dd6da8378ebc374efee407cb
x/btclightclient/types/types.go	4c2a0dcb746fede3b4dce658a7d41811ea8dc7ab
x/btclightclient/types/utils.go	0d472392e370e04ce51c0933a742a169feeeaf36
x/btclightclient/types/work.go	375b03a9d879af4344f011565b7fc9b6cb06cf8b
x/btclightclient/genesis.go	c295f76a8855cf1b2bf9df601219bcec7521ade0
x/btclightclient/module.go	470370f4304a4f6e71a823a33e799d2b96a94ceb
x/btcstaking/client/cli/query.go	b6be8442a6540fed597dd8de28255f7317a2d860
x/btcstaking/client/cli/tx.go	275074212832a90dc84e797dff47a0ba426eb1b2
x/btcstaking/client/cli/utils.go	ae47e4c32436a3bf0593aa2180cc5999daa7fea5
x/btcstaking/keeper/btc_staking_record.go	b253edf3768cc0866d84193b3d4d157a9d20c9d6
x/btcstaking/keeper/grpc_query.go	18037960d49b4bc70a3e08ff2a48449c5f6bbae4
x/btcstaking/keeper/keeper.go	72b570ed8b5fc71a3b63bbaf1332228682611eb8
x/btcstaking/keeper/msg_server.go	a985a39fc7ac9cbef94cb7db1f785cccb958df09
x/btcstaking/keeper/params.go	719fba6cb319874c0648ce526f889e2b6bf069bb
x/btcstaking/keeper/utils.go	940c7552388f6c8d4180812e210b952cf4613bf1
x/btcstaking/types/codec.go	9c796daeb19bb9c855bbce9607deb2c9bff294b3
x/btcstaking/types/errors.go	31be48ac2ab575cb83e468aa063b6d5f04e10e8c
x/btcstaking/types/event.pb.go	2f51dea129798cfde36bb9139c0cc58632ab8eab
x/btcstaking/types/events.go	23d8daef33a2b0b8cd60908b9c23f9b8e29e3ef0
x/btcstaking/types/expected_keepers.go	3682b360cf739eb48fa3e2bcf7fa17c81b8fa8ab
x/btcstaking/types/genesis.go	2d035fcb4d8ccb3217294dc9e97d7fd986eddb60
x/btcstaking/types/genesis.pb.go	05874147134ae416de67388cef97fc3cfc2c6fd6
x/btcstaking/types/keys.go	276300702e5e056e1d2ae08f2472d1580eaab3ae
x/btcstaking/types/msg.go	4289a622f2eb820723fca73a2ff8ce6237486bc6

x/btcstaking/types/params.pb.go	16ba9821118a78f15e88c0f34e1248658df0d646
x/btcstaking/types/query.pb.go	0e8b137a8a0aab8f610b5b4e8437fbb6adb24b42
x/btcstaking/types/query.pb.gw.go	83eab0bc535e09d5a5847bd2d487f8272ecaff50
x/btcstaking/types/staking_record.pb.go	9b6ae126db437395ada7bcc5e0134418f3d41501
x/btcstaking/types/tx.pb.go	b391596cb44eea7b6751a4071bc461d92b21000a
x/btcstaking/types/types.go	bce48f6e6b681dbb12f38d001835615347789221
x/btcstaking/module.go	bcf507619a7019064d31ed746fb4d9679d639f31
x/fee/client/cli/query.go	847497f6cf9adc65560993ee6afe65e263364c5c
x/fee/keeper/grpc_query.go	045146714bf1abe24cca74787939f6a2b8014c6b
x/fee/keeper/keeper.go	89fafa13293922053ce414d734549a3f230aceb4
x/fee/keeper/msg_server.go	c5000ba04711a732d294a39f65fcf97011605461
x/fee/keeper/tx_fee_checker.go	1c5160f2a9f875099f7dfdce2eb22e486176ff22
x/fee/types/codec.go	31d5ee927f562905917291c49349b316f95c1fef
x/fee/types/errors.go	f95448041f0b253168f9069879fe4ee084e1a4ac
x/fee/types/genesis.pb.go	44716de811f21e08edab4465dff8560173b25269
x/fee/types/keys.go	a4eca68cb1c475a0d7f761e3487bd49f25e94264
x/fee/types/msgs.go	9798717875d6d33d3f6c095309f439263b05d4f4
x/fee/types/params.pb.go	4552ebb651ed8a235a93e0a222cbd8456496b108
x/fee/types/query.pb.go	d89ecd20be1a66373b8b2b123409e33738c10974
x/fee/types/query.pb.gw.go	a0459f31b566717a56c43f4517c37c52be0075c0
x/fee/types/tx.pb.go	2e67c4399fd1fe9328400fabba5465abc042dfb4
x/fee/genesis.go	cefab28ddaac06ae73eb5a511e798c5458a5ebd5
x/fee/module.go	4860203681891499fa9261f6fcfa4ff6685b881a