SALUS SECURITY

APR 2025

CODE SECURITY ASSESSMENT

XONE

Overview

Project Summary

- Name: Xone - chain
- Platform: EVM-compatible chains
- Language: Go
- Repository:
    - https://github.com/hello-xone/xone_chain
- Audit Range: See Appendix - 1

Project Dashboard

Application Summary

Name	Xone - chain
Version	v2
Type	Go
Dates	Apr 07 2025
Logs	Mar 31 2025; Apr 07 2025

Vulnerability Summary

Total High-Severity issues	1
Total Medium-Severity issues	1
Total Low-Severity issues	0
Total informational issues	1
Total	3

Contact

E-mail: support@salusec.io

Risk Level Description

High Risk	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.
Medium Risk	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.
Low Risk	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.
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or defense in depth.

Content

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), or Email (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	The HTTP/2 protocol in Golang prior to 1.21.9 is vulnerable to DoS	High	Configuration	Resolved
2	Potential chain halt due to unhandled errors	Medium	Configuration	Resolved
3	Incomplete keeper interface definitions	Informational	Business Logic	Resolved

2.2 Notable Findings

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

1. 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 go 1.19

An attacker can cause the HTTP/2 endpoint to read arbitrary amounts of header data by sending an unlimited number of CONTINUATION frames. This causes 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.

References:

https://kb.cert.org/vuls/id/421644

https://www.cve.org/CVERecord?id=CVE-2023-45288

Recommendation

1. Upgrade to a newer version. For instance, 1.22.2 which has fixes of other security issues as well (or at least 1.21.9).

2. Run go mod tidy to ensure compatibility

Status

The team has resolved this issue in commit 596c2bd.

2. Potential chain halt due to unhandled errors

Severity: Medium	Category: Configuration
Target: - Cosmos SDK	

Description

The xone blockchain project uses a custom fork of Cosmos SDK (github.com/huione-labs/cosmos-sdk v0.47.5-evm-cointype) that is vulnerable to ISA-2025-002, a known security issue in the x/group module. This vulnerability could allow malicious actors to submit proposals that trigger errors in the module's end blocker, potentially resulting in a complete chain halt.

Analysis of the project's dependencies shows that the codebase is using a vulnerable version of Cosmos SDK (v0.47.5-based fork), which falls within the affected range (≤ v0.47.16).

The vulnerability specifically occurs in the doTallyAndUpdate function of the x/group module's keeper. When errors occur during proposal tallying, these errors are not properly handled, which can lead to chain halts.

Recommendation

Apply the security patch from Cosmos SDK v0.47.17 to your forked repository.

Status

The team has resolved this issue in commit 596c2bd.

2.3 Informational Findings

3. Incomplete keeper interface definitions

Severity: Informational	Category: Business Logic
Target: - expected_keepers.go	

Description

The expected_keepers.go file in the Xone module contains placeholder interfaces for interacting with other Cosmos SDK modules. These interfaces currently lack method definitions, preventing proper module integration. Only AccountKeeper and BankKeeper have minimal method signatures, but most interfaces (AuthzKeeper, MintKeeper, etc.) are empty placeholders.

This issue prevents the module from accessing essential blockchain functionality like token transfers, staking operations, and parameter management, despite these dependencies being initialized in keeper.go.

Recommendation

Define Required Methods: Add concrete method signatures for each keeper interface, only methods the Xone module will actually use:

```
type BankKeeper interface {
    SpendableCoins(ctx sdk.Context, addr sdk.AccAddress) sdk.Coins
    SendCoins(ctx sdk.Context, fromAddr sdk.AccAddress, toAddr sdk.AccAddress, amt
sdk.Coins) error
}
```

Status

The team has resolved this issue in commit 596c2bd.

Appendix

Appendix 1 - Files in Scope

This audit covered the following files in commit 00d9133:

File	SHA-1 hash
xone/genesis.go	3124460b9dc8e30a3a7d90154bf100f62f01bef0
xone/module.go	0b3f0fc16730aa604d8378b2e70a5fb0ecddaaa1
xone/module_simulation.go	991c55617c298e4193873dc8a0346b4c354f7ce8
xone/keeper/keeper.go	67d99389d5b26ade944b47505d0c0eb1a705386d
xone/keeper/msg_server.go	4309bbb3e2c63cd483a71edc488f43357f2084ac
xone/keeper/params.go	5b505d36b31f7a18cd86901212301b687949897d
xone/keeper/query.go	99c3521c0b4a58a83e58a0e23d42965aa7216cf4
xone/keeper/query_params.go	0652d2878b06456414594baab4b8cd181597458b
xone/simulation/helpers.go	3c681e4052b98cd7ef44607866ef0b3419f0f8cd
xone/types/codec.go	66ea2061954c27af23adef6823d2632e7974fdc3
xone/types/errors.go	49b1ef69c7165a281d2ae89ca95594e6382fb707
xone/types/expected_keepers.go	ad4ef7724ff4c3ffcac1e88a473b1866f7d4512a
xone/types/genesis.go	a7058f6a7fccbfc42b59190e082a48a569e666fc
xone/types/genesis.pb.go	be6b5851165fb838f4bb8b6fd1ccc9797ef96c23
xone/types/keys.go	ecc673a96fb868e4942a873ab47fb7b75a79e70c
xone/types/params.go	a53d89392a6e023aaca776216b47969dfe737368
xone/types/params.pb.go	5032cc83cc5f379acb88135ed695e86d1077a2ad
xone/types/query.pb.go	51a209c86171ae46509da5cc43a719ec89c76a46
xone/types/query.pb.gw.go	162486538469b89db0ec0b9e6c043d22d3ed9a69
xone/types/tx.pb.go	9bffecfa54d60a7068f98c065687170ea088e365
xone/types/types.go	4c2a0dcb746fede3b4dce658a7d41811ea8dc7ab
xone/client/cli/query.go	1c3d5e68d51af6d6773348016fc2f960648721af
xone/client/cli/query_params.go	54da5c4ffdad26ebbb4cf5ffc3a98b8258036f94

xone/client/cli/tx.go	b63c6742abcf43fa81657b88d1de4ccee7ee0be9