



*Blockchain Security Assessment*

# Symphony

Verified on 07/10/2023

## SUMMARY

Project

Symphony

CHAIN

Symphony

METHODOLOGY

Manual &amp; Automatic Analysis

FILES

Single

DELIVERY

07/10/2023

TYPE

Blockchain Audit



4

0

0

0

0

0

4

Total Findings

Resolved

Critical

Major

Medium

Minor

Informational

 0 Critical

An exposure that can affect the contract functions in several events that can risk and disrupt the contract

 0 Major


An exposure that can affect the outcome when using the contract that can serve as an opening in manipulating the contract in an unwanted manner

 0 Medium

An opening that could affect the outcome in executing the contract in a specific situation

 0 Minor

An opening but doesn't have an impact on the functionality of the contract

 4 Informational

An opening that consists information but will not risk or affect the contract

**STATUS**
 **AUDIT PASSED**

# TABLE OF CONTENTS | Symphony

## | Summary

Project Summary  
Findings Summary  
Disclaimer  
Scope of Work  
Auditing Approach

## | Project Information

Token/Project Details  
Inheritance Graph  
Call Graph

## | Findings

Issues  
SWC Attacks  
CW Assessment  
Fixes & Recommendation  
Audit Comments

## DISCLAIMER | Symphony

**ContractWolf** audits and reports should not be considered as a form of project's "Advertisement" and does not cover any interaction and assessment from "Project Contract" to "External Contracts" such as PancakeSwap, UniSwap, SushiSwap or similar.

**ContractWolf** does not provide any warranty on its released report and should not be used as a decision to invest into audited projects.

**ContractWolf** provides a transparent report to all its "Clients" and to its "Clients Participants" and will not claim any guarantee of bug-free code within its Client Files.

**ContractWolf's** presence is to analyze, audit and assess the Client's Files to find any underlying risk and to eliminate any logic and flow errors within its code.

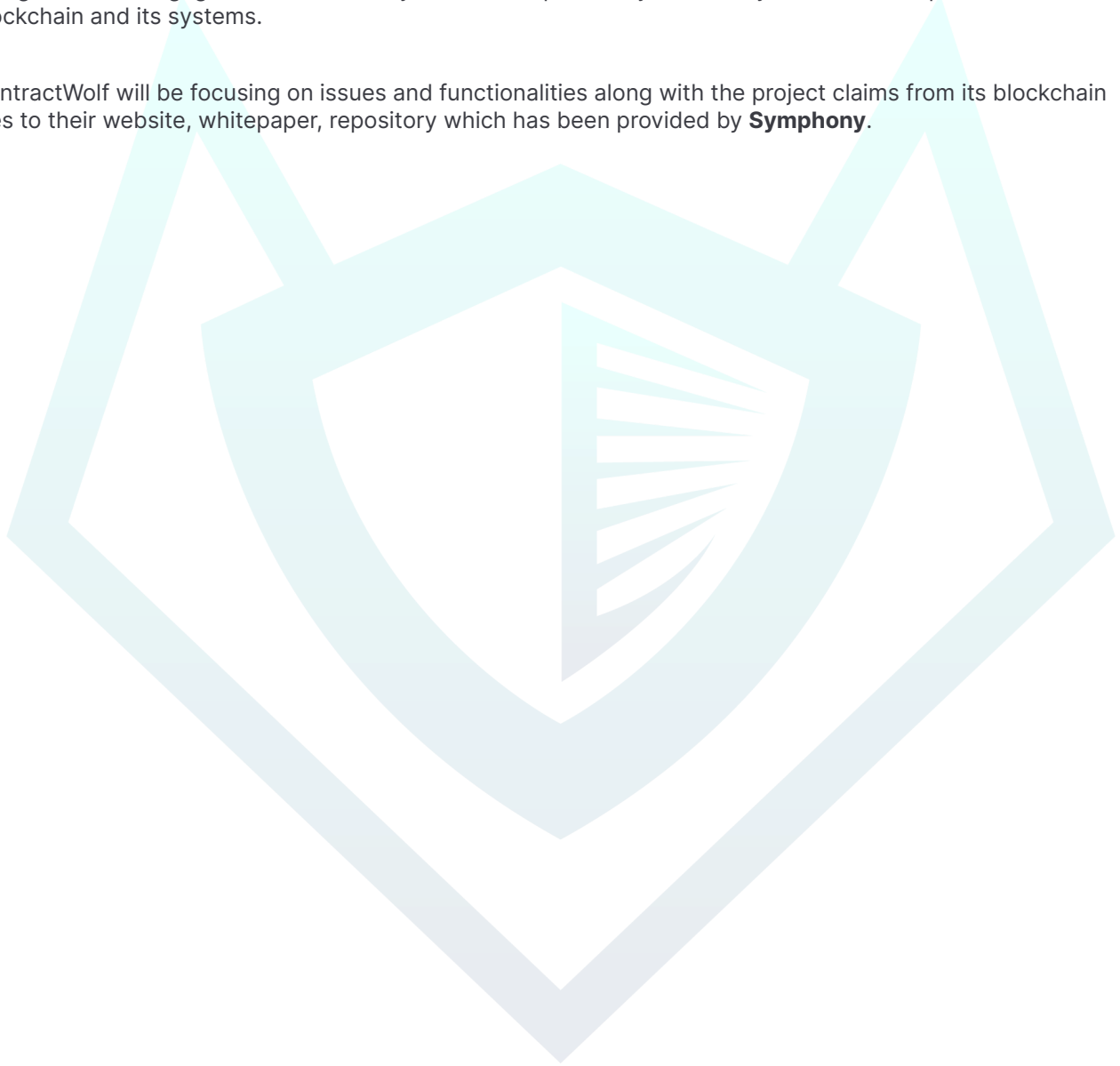
*Each company or project should be liable to its security flaws and functionalities.*

## SCOPE OF WORK | Symphony

**Symphony** team has agreed and provided us with the files that need to be tested (*Github, BSCscan, Etherscan, Local files etc*). The scope of audit is the main blockchain files.

The goal of this engagement is to identify if there is a possibility of security flaws in the implementation of its blockchain and its systems.

ContractWolf will be focusing on issues and functionalities along with the project claims from its blockchain files to their website, whitepaper, repository which has been provided by **Symphony**.



## AUDITING APPROACH | Symphony

Every line of code along with its functionalities will undergo manual review to check for security issues, quality of logic and contract scope of inheritance. The manual review will be done by our team that will document any issues that they discovered.

### METHODOLOGY

The auditing process follows a routine series of steps :

1. Code review that includes the following :
  - Review of the specifications, sources and instructions provided to ContractWolf to make sure we understand the size, scope and functionality of the Blockchain.
  - Manual review of code. Our team will have a process of reading the code line-by-line with the intention of identifying potential vulnerabilities, underlying and hidden security flaws.
2. Testing and automated analysis that includes :
  - Testing the function with common test cases and scenarios to ensure that it returns the expected results.
3. Best practices and ethical review. The team will review the contract with the aim to improve efficiency, effectiveness, clarifications, maintainability, security and control within the blockchain.
4. Recommendations to help the project take steps to eliminate or minimize threats and secure the blockchain.

## TOKEN DETAILS | Symphony



Symphony is a chain dedicated to fully decentralized real-world assets. Its dual-elasticity structure allows for automated rebalancing from any drop in value.

### SOURCE

Source

<https://github.com/Orchestra-Labs/symphony-osmosis>

# FINDINGS | Symphony



4

0

0

0

0

0

4

Total Findings

Resolved

Critical

Major

Medium

Minor

Informational

This report has been prepared to state the issues and vulnerabilities for Symphony through this audit. The goal of this report findings is to identify specifically and fix any underlying issues and errors

Title	File	Severity	Status
Error Handling	config.go <i>app/params/config.go</i>	Informational	● Pending
Potential Injection	tx.go <i>x/pool-incentives/client/cli/tx.go</i>	Informational	● Pending
Error Handling	genesis.go <i>x/mint/keeper/genesis.go</i>	Informational	● Pending
Potential nil Issue(s)	keeper.go <i>x/mint/keeper/keeper.go</i>	Informational	● Pending



## FIXES & RECOMMENDATIONS

### Security Analysis | [config.go](#) [app/params/config.go](#)

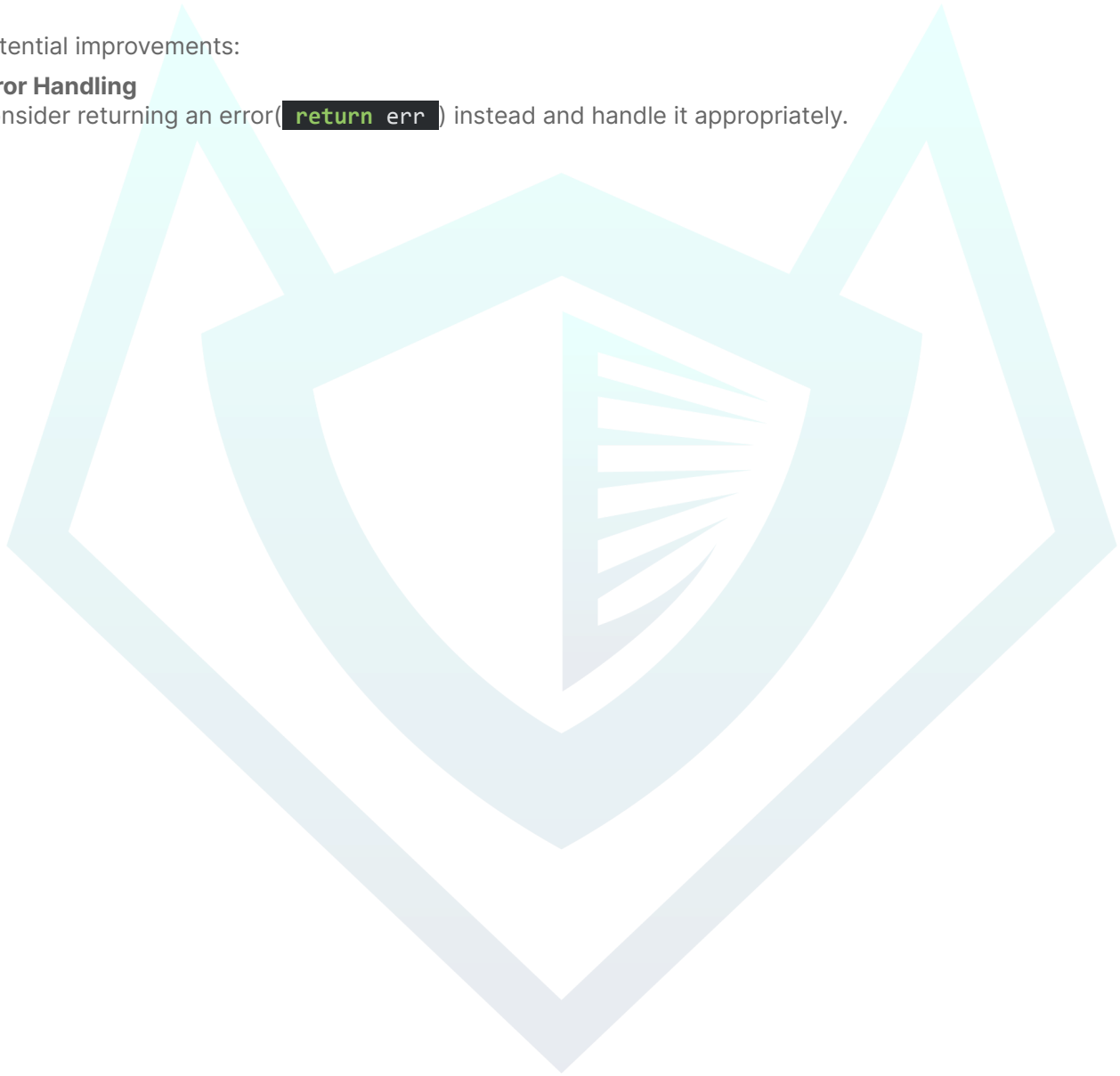
Security vulnerabilities:

The use of `panic` in `RegisterDenoms` for handling errors can cause the entire application to crash.

Potential improvements:

#### Error Handling

Consider returning an error(`return err`) instead and handle it appropriately.



## Security Analysis | tx.go

*x/pool-incentives/client/cli/tx.go*

### Security vulnerabilities:

Malicious actors could potentially submit proposals with invalid `gaugeIDs` or `weights` that might cause unexpected behavior.

### Potential improvements:

#### **Input validation**

Consider adding validation for `gaugeIDs` (eg. ensuring they exist within a defined set of valid gauges) and `weights` (eg. restricting them to a valid range)

### Security Considerations:

There aren't any immediate security concerns in *tx.go*. However, it's generally good practice to validate user(s) input beyond basic format checks to prevent potential injection attacks.

## Security Analysis | tx.go

*x/valset-pref/client/cli/tx.go*

***No issues found***

Investigated the ff :

*Overall code performance, errors & input validations*



## Security Analysis | genesis.go

*x/mint/keeper/genesis.go*

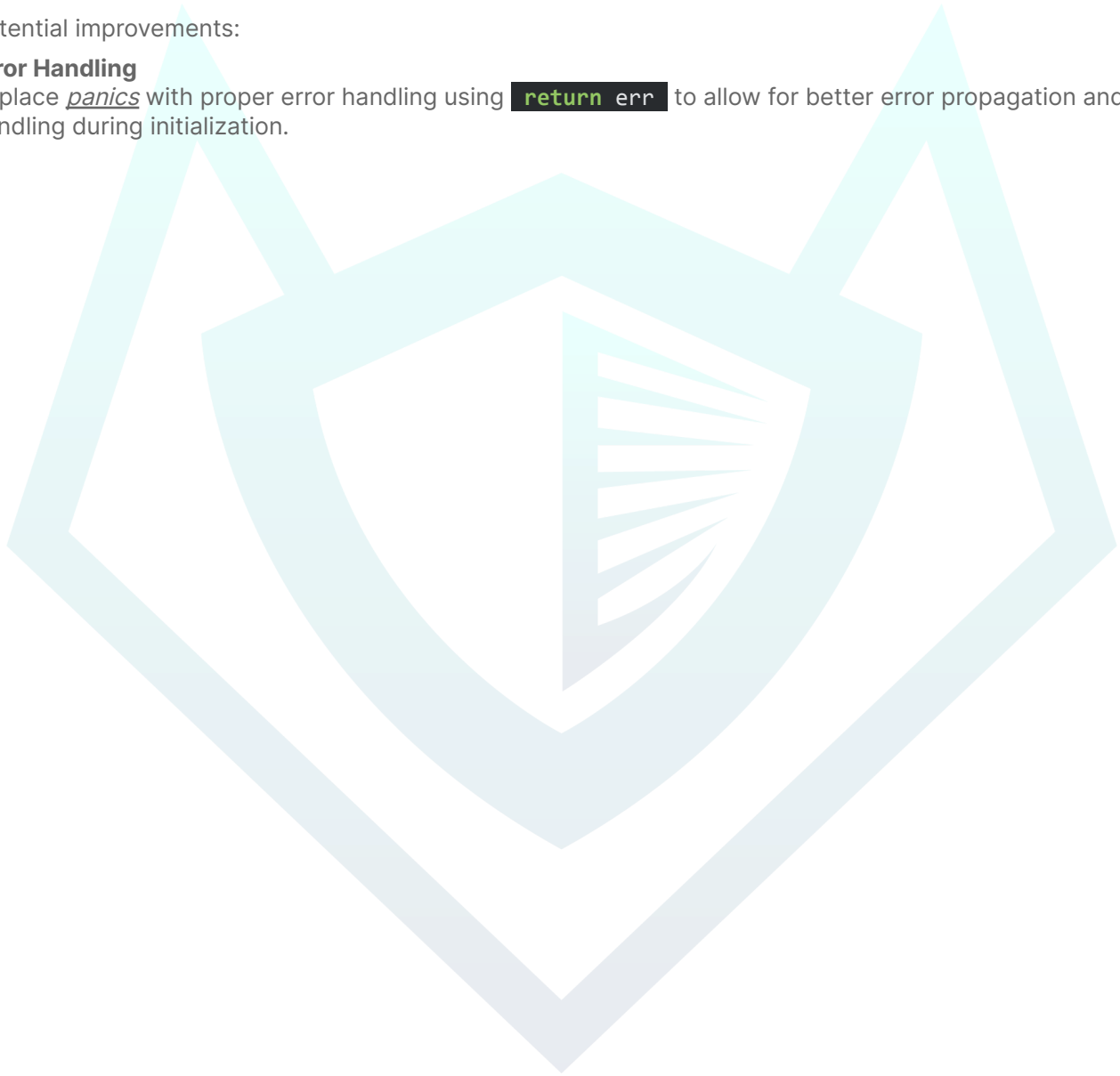
Security vulnerabilities:

The `createDeveloperVestingModuleAccount` function calls `panic` in case of an error. Consider returning an error instead to allow for proper handling during initialization.

Potential improvements:

### Error Handling

Replace panics with proper error handling using `return err` to allow for better error propagation and handling during initialization.



## Security Analysis | keeper.go

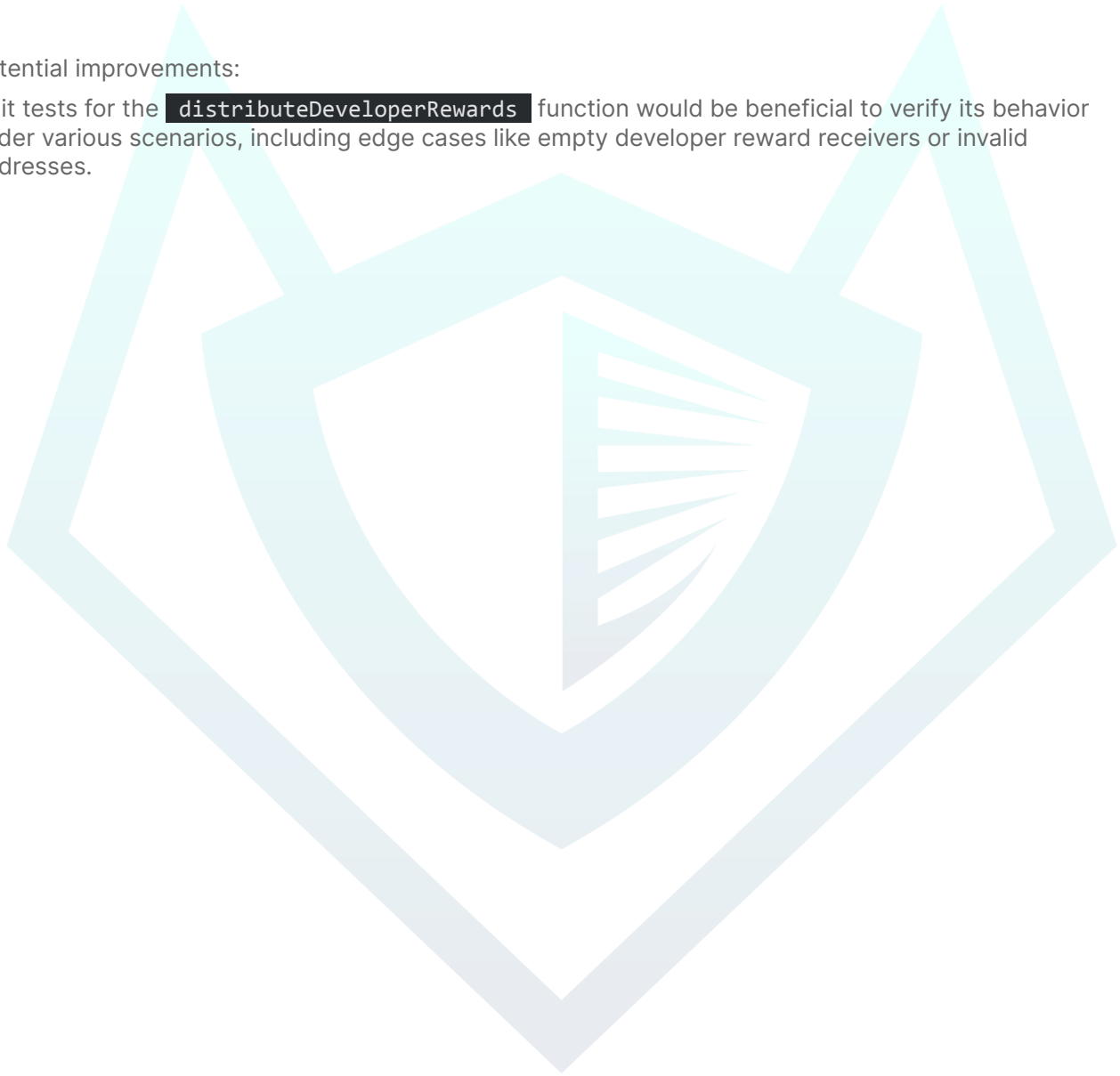
*x/mint/keeper/keeper.go*

### Security vulnerabilities:

Consider adding checks for empty `developerRewardsReceivers` before iterating to avoid potential nil pointer issues.

### Potential improvements:

Unit tests for the `distributeDeveloperRewards` function would be beneficial to verify its behavior under various scenarios, including edge cases like empty developer reward receivers or invalid addresses.



## Security Analysis | query.go

*x/mint/client/cli/query.go*

***No issues found***

Investigated the ff :

*Overall code performance, errors & input validations*



## AUDIT COMMENTS | Symphony

- This **Blockchain Audit** is exclusively for the project **Symphony**
- **Symphony** may or may not follow the recommendations with the findings.
- The files from */symphony-osmosis* folder over the repository may be updated upon deployment and ContractWolf holds no warranty over the changes that will be made in the future.





# **CONTRACTWOLF**

**Blockchain Security - Smart Contract Audits**