// HALBORN

Mars Protocol - Periphery Delegator

CosmWasm Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **October 24th, 2022 - October 26th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	10/24/2022	Elena Maranon
0.2	Draft Version	10/26/2022	Elena Maranon
0.3	Draft Review	10/28/2022	Gabi Urrutia
1.0	Remediation Plan	11/04/2022	Elena Maranon
1.1	Remediation Plan Review	11/09/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Luis Quispe Gonzales	Halborn	Luis.QuispeGonzales@halborn.com
Elena Maranon	Halborn	Elena.Maranon@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Mars Protocol engaged Halborn to conduct a security audit on their smart contracts beginning on October 24th, 2022 and ending on October 26th, 2022. The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided three days for the engagement and assigned a full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some security risks that were partially addressed by the Mars Protocol team. The main one is the following:

- Verify the delegator ending time configured during instantiation to ensure it is not set in the past.

1.3 TEST APPROACH & METHODOLOGY

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.
1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

10 - CRITICAL
9 - **8** - HIGH

7 - 6 - MEDIUM

5 - 4 - LOW

3 - 1 - VERY LOW AND INFORMATIONAL

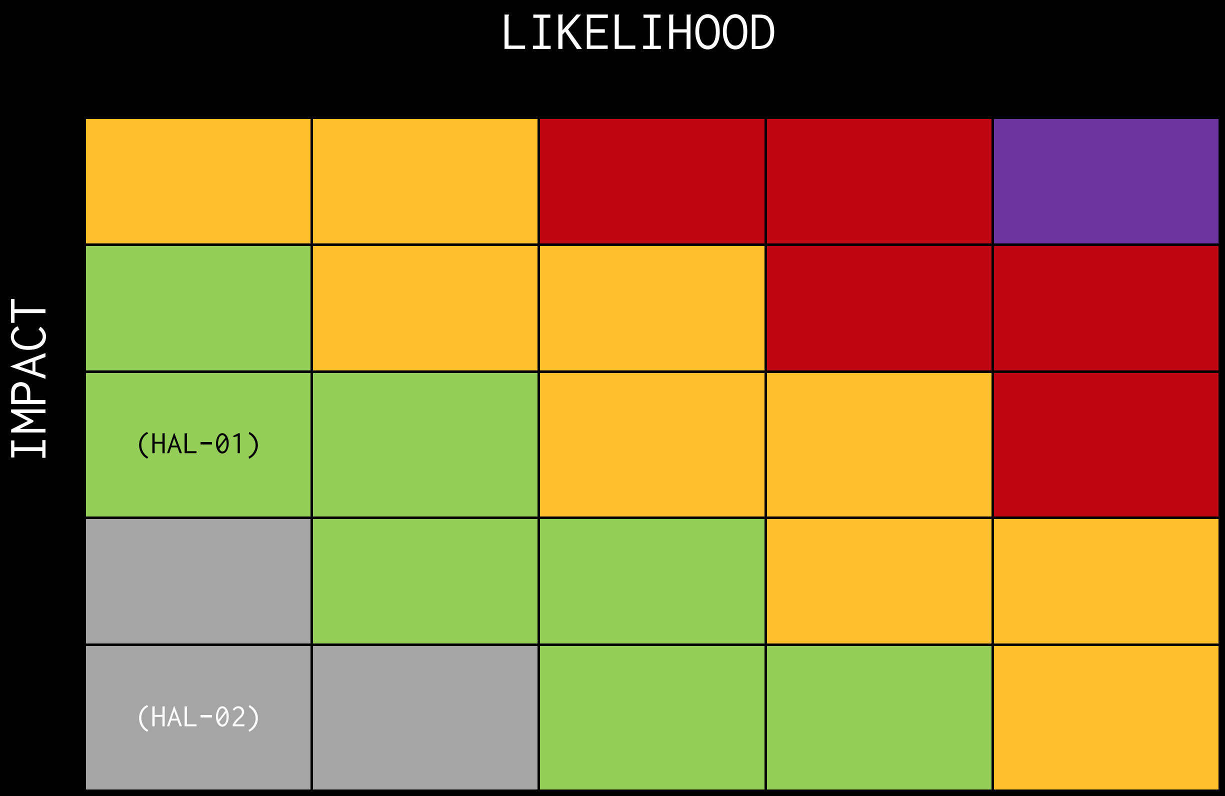
1.4 SCOPE

1. CosmWasm Smart Contracts
   (a) Repository: periphery
   (b) Commit ID: fac9bd73d80d1e39ca23875e4a46287c82965461
   (c) Contracts in scope:
      i. delegator
   (d) Packages in scope:
      i. types

Out-of-scope: External libraries and financial related attacks

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	1	1



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) ENDING TIMESTAMP DO NOT HAVE A MINIMUM THRESHOLD	Low	RISK ACCEPTED
(HAL-02) OVERFLOW CHECKS NOT SET FOR CONTRACT PROFILE RELEASE	Informational	SOLVED - 11/04/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) ENDING TIMESTAMP DO NOT HAVE A MINIMUM THRESHOLD - LOW

Description:

In delegator contract, the Config struct used to define the length of the original delegation period within the instantiate function does not validate that the ending timestamp (ending_time) is greater than the current timestamp.

Taking into consideration that the purpose of this contract is to be deployed at launching in order to avoid that a post-genesis validator could hijack the network after an airdrop, if the **ending_time** is miss-configured, the refund could be executed at any moment and the contract would fail in its functionality.

Code Location:

Fragment of init function from delegator contract:

Listing 1: contracts/delegator/src/execute.rs (Line 16)

```
13 pub fn init(deps: DepsMut, info: MessageInfo, cfg: Config) ->
↳ Result<Response, ContractError> {
14     // We don't implement a validity check of the ending time.
15     // The deployer must make sure to provide a valid value.
16     CONFIG.save(deps.storage, &cfg)?;
17
18     let amount = must_pay(&info, &cfg.bond_denom)?;
19     let msgs = get_delegation_msgs(&deps.querier, amount.u128(), &
↳ cfg.bond_denom)?;
20
21     Ok(Response::new().add_messages(msgs))
22 }
```

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

It is recommended to update the logic of instantiate function mentioned above to validate that ending timestamp is greater than current timestamp.

Remediation Plan:

RISK ACCEPTED: The Mars Protocol team accepted the risk of this finding.

3.2 (HAL-02) OVERFLOW CHECKS NOT SET FOR CONTRACT PROFILE RELEASE - INFORMATIONAL

Description:

Although the overflow-checks parameter is set to **true** in profile.release and implicitly applied to all contracts and packages in the workspace, it is not explicitly enabled in **Cargo.toml** for delegator contract, which could have unexpected consequences if the project is refactored.

Code Location:

Listing 2: Resources affected

```
contracts/delegator/Cargo.toml
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended that you explicitly enable overflow checks on each individual contract and package. That measure helps when the project is refactored to avoid unintended consequences.

Remediation Plan:

SOLVED: The Mars Protocol team solved this issue in the commit ID 0150502fe15a8fd15c8d2436ca6395e613606fe5.

THANK YOU FOR CHOOSING

// HALBORN