// HALBORN

Mars Protocol Core Contracts (Updated code)

CosmWasm Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **February 14th, 2022 - February 18th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY 2

CONTACTS 2

1 EXECUTIVE OVERVIEW 3

1.1 AUDIT SUMMARY 4

1.2 TEST APPROACH & METHODOLOGY 5

RISK METHODOLOGY 5

1.3 SCOPE 7

2 ASSESSMENT SUMMARY & FINDINGS OVERVIEW 8

3 FINDINGS & TECH DETAILS 9

3.1 (HAL-01) MULTIPLE CONSEQUENCES WHEN INTERACTING WITH UPPER-CASE ADDRESSES - HIGH 11

Description 11

Code Location 11

Risk Level 11

Recommendation 12

Remediation plan 12

3.2 (HAL-02) EQUIVALENCE BETWEEN MARS AND XMARS TOKENS CAN BE BROKEN - LOW 13

Description 13

Code Location 13

Risk Level 14

Recommendation 14

Remediation plan 14

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	02/14/2022	Luis Quispe Gonzales
0.2	Document Updates	02/16/2022	Luis Quispe Gonzales
0.3	Draft Version	02/18/2022	Luis Quispe Gonzales
0.4	Draft Review	02/18/2022	Gabi Urrutia
1.0	Remediation Plan	02/21/2022	Luis Quispe Gonzales
1.1	Remediation Plan Review	02/22/2022	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

EXECUTIVE OVERVIEW

1.1 AUDIT SUMMARY

Mars Protocol engaged Halborn to conduct a security assessment on CosmWasm smart contracts beginning on February 14th, 2022 and ending February 18th, 2022.

The security engineers involved on the audit are blockchain and smart-contract security experts with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to achieve the following:

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

In summary, Halborn identified some improvements to reduce the likelihood and impacts of the risks, which were accepted by Mars team. The main ones are the following:

- Update the logic of some functions in mars-red-bank contract to turn addresses into lower case.
- Update and store the amount of Mars tokens owned by mars-staking contract in its storage when staking / unstaking.

External threats, such as financial related attacks, oracle attacks, and inter-contract functions and calls should be validated for expected logic and state.

1.2 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual review of the code and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of the smart contract audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of smart contracts and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture, purpose, and use of the platform.
- Manual code read and walkthrough.
- Manual assessment of use and safety for the critical Rust variables and functions in scope to identify any contracts logic related vulnerability.
- Fuzz testing (Halborn custom fuzzing tool)
- Checking the test coverage (cargo tarpaulin)
- Scanning of Rust files for vulnerabilities (cargo audit)

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.

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.3 SCOPE

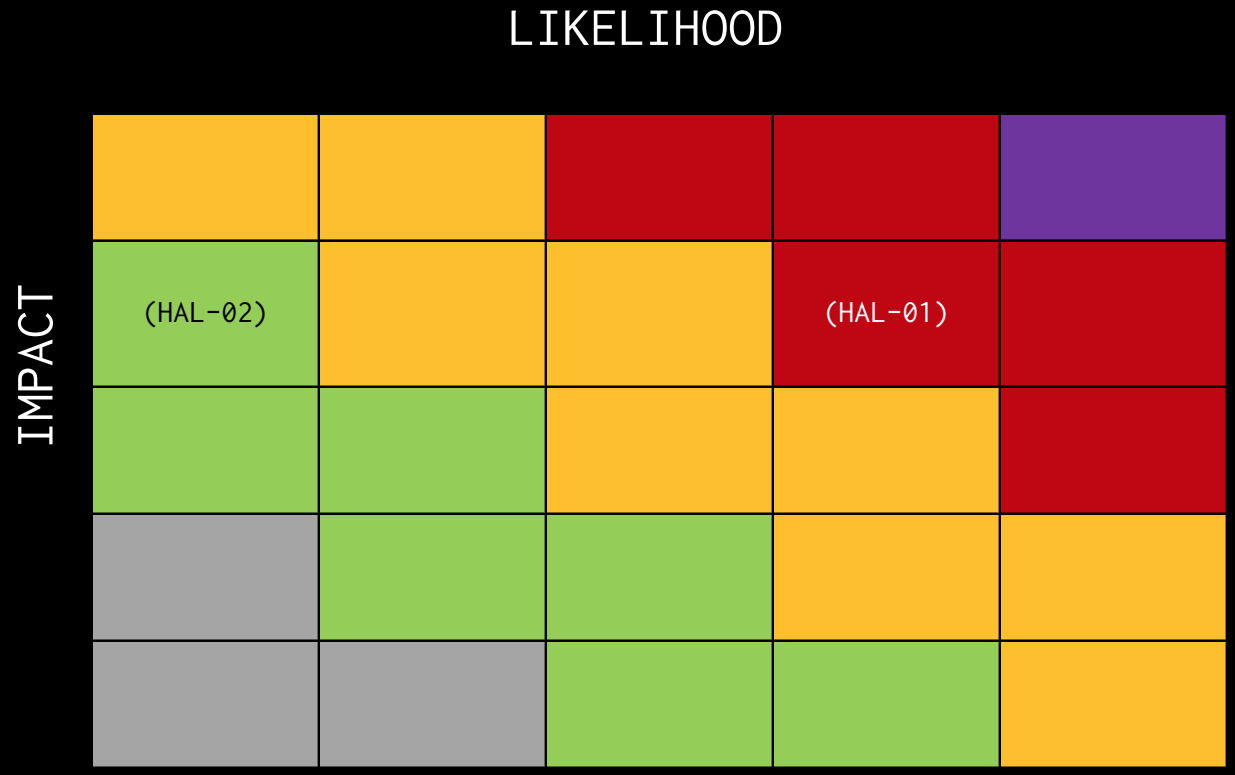
1. CosmWasm Smart Contracts
   (a) Repository: mars-core
   (b) Commit ID: d08152d5fc0294635c57b8bc51fe0962c79e4363
   (c) Contracts in scope:
      i. mars-council
      ii. mars-red-bank
      iii. mars-vesting

It is worth noting that the results of this audit are a complement to the information provided in a previous report for the security audit performed to the codebase with commit id fcaa6ffe918a0890a1a6c57f26edb6f8feb25633.

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

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	1	0	1	0



EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) MULTIPLE CONSEQUENCES WHEN INTERACTING WITH UPPER-CASE ADDRESSES	High	FUTURE RELEASE
(HAL-02) EQUIVALENCE BETWEEN MARS AND XMARS TOKENS CAN BE BROKEN	Low	RISK ACCEPTED

FINDINGS & TECH DETAILS

3.1 (HAL-01) MULTIPLE CONSEQUENCES WHEN INTERACTING WITH UPPER-CASE ADDRESSES - HIGH

Description:

When users deposit, withdraw, borrow or repay in **contracts/mars-red-bank/src/contract.rs** using on_behalf_of / recipient_address address in upper case (e.g.: **TERRA1KG. . . XNL8**), the following consequences occur:

- Deposited tokens cannot be used to borrow.
- Recipient cannot use his withdrawn tokens.
- Recipient cannot use his borrowed tokens.
- Repaying will throw error messages.

These issues happen because the values are stored in contract's storage with the upper case address as a key, which creates conflicts when values are loaded using info.sender as a key, which is always in lower case (e.g.: **terra1kg. . . xnl8**).

Code Location:

Listing 1: Resources affected

```
mars-red-bank: execute_deposit (on_behalf_of)
mars-red-bank: execute_withdraw (recipient_address)
mars-red-bank: execute_borrow (recipient_address)
mars-red-bank: execute_repay (on_behalf_of)
```

Risk Level:

Likelihood - 4
Impact - 4

Recommendation:

Update the logic of functions mentioned above to turn recipient addresses into lower case.

Remediation plan:

PENDING: The Mars team stated that in the short term they would analyze how to address this issue comprehensively on the protocol.

3.2 (HAL-02) EQUIVALENCE BETWEEN MARS AND XMARS TOKENS CAN BE BROKEN - LOW

Description:

execute_create_allocation function in **contracts/mars-vesting/src/contract.rs** relies on the equivalence between Mars and xMars tokens remains true, i.e.: 1 Mars = 1 xMars. However, this equivalence can be broken if a malicious user transfers Mars tokens to **mars-staking** contract.

As a consequence, execute_create_allocation function cannot be called, i.e.: a Denial-of-Service (DoS) that could even be permanent. It is worth noting that the likelihood for this to happen is low because allocations are created at one of the phases of protocol launch where Mars tokens have not been distributed to anyone but the multisig wallet.

Code Location:

Listing 2: contracts/mars-staking/src/contract.rs (Lines 544-548)

```
535 fn get_staking_tokens_info(
536     deps: Deps,
537     env: &Env,
538     config: &Config,
539     global_state: &GlobalState,
540     mars_to_deduct: Uint128,
541 ) -> StdResult<StakingTokensInfo> {
542     let (mars_token_address, xmars_token_address) =
            get_token_addresses(deps, config)?;
543
544     let total_mars_in_staking_contract = cw20_get_balance(
545         &deps.querier,
546         mars_token_address.clone(),
547         env.contract.address.clone(),
548     )?
549     .checked_sub(mars_to_deduct)?;
```

Risk Level:

Likelihood - 1
Impact - 4

Recommendation:

Fix the logic of get_staking_tokens_info function to update and store the amount of Mars tokens owned by **mars-staking** contract in its storage when staking / unstaking, instead of relying on Mars token balance.

Remediation plan:

RISK ACCEPTED: The Mars team accepted the risk for this finding. They also stated that the check in **mars-vesting** contract is a guarantee that they won't create more allocations for the contract and that the voting power given to this Mars token is fair.

THANK YOU FOR CHOOSING

// HALBORN