// HALBORN

Astroport.fi Factory and Generator

CosmWasm Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **April 25th, 2022 - April 29th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	04/25/2022	Connor Taylor
0.2	Document Updated	04/28/2022	Connor Taylor
0.3	Document Updated	05/10/2022	Jakub Heba
0.4	Draft Version	05/11/2022	Connor Taylor
0.5	Draft Review	05/12/2022	Gabi Urrutia
1.0	Remediation Plan	06/01/2022	Jakub Heba
1.1	Remediation Plan Review	06/02/2022	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
Connor Taylor	Halborn	Connor.Taylor@halborn.com
Jakub Heba	Halborn	jakub.heba@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Astroport.fi engaged Halborn to conduct a security audit on their smart contracts beginning on April 25th and ending on April 29th. The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided one week for the engagement and assigned two full-time security engineers to audit the security of the smart contract. The security engineers 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:

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

In summary, Halborn identified some improvements to reduce the likelihood and impact of risks, which were partially addressed by Astroport team. The main improvements highlighted in this report are:

- Reduce the Generator contract size, to be able to deploy it successfully.
- Ensure functionality is implemented to mitigate the impact of boost abuse.

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

1.3 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

1.4 SCOPE

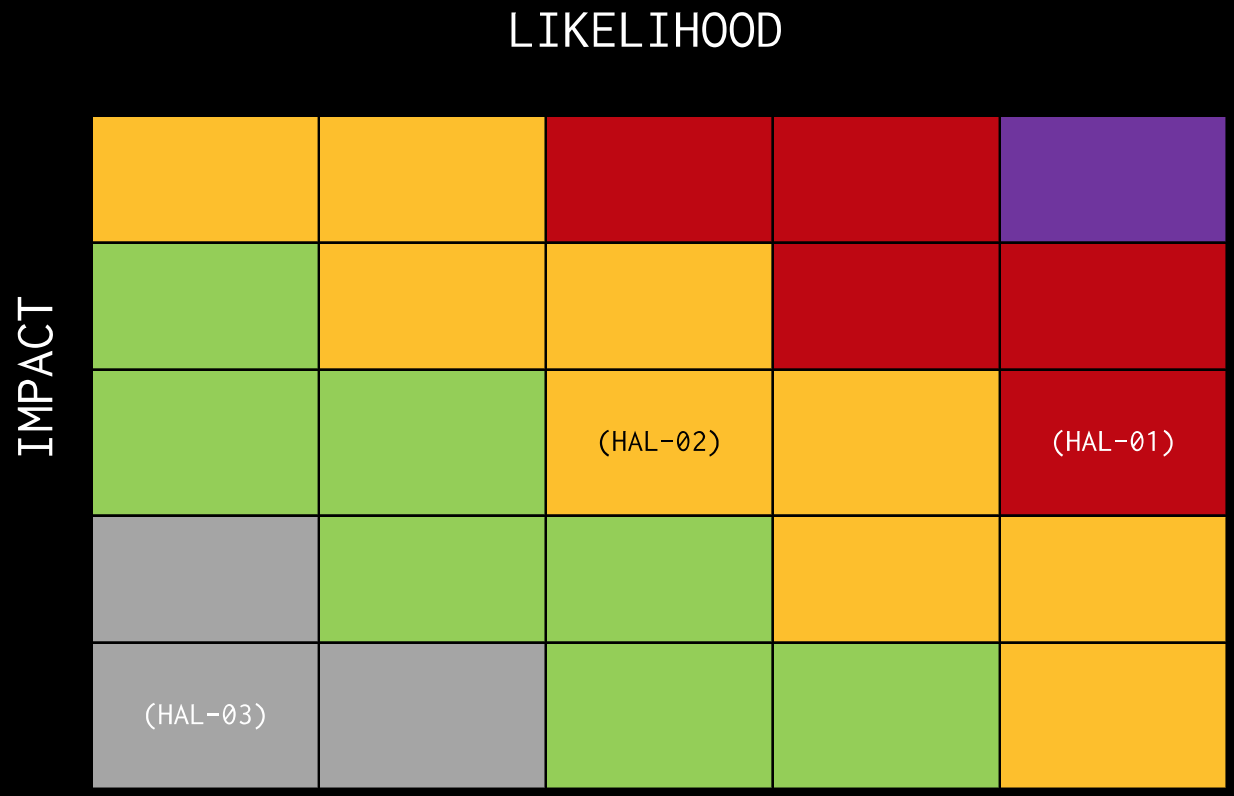
Code repository: astroport-core

1. CosmWasm Smart Contracts
    (a) Commit ID: 4a0a4fc619a7549e1f347727d57e17522719f3fb
    (b) Contracts in scope:
        i. Factory
        ii. Tokenomics/Generator

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

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	1	1	0	1



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) MAXIMUM SIZE OF COMPILED CONTRACT EXCEEDED	High	SOLVED - 05/05/2022
(HAL-02) LACK OF BAN FUNCTIONALITY WITHIN BOOSTS	Medium	RISK ACCEPTED
(HAL-03) OVERFLOW CHECKS NOT SET FOR PROFILE RELEASE	Informational	ACKNOWLEDGED

FINDINGS & TECH DETAILS

3.1 (HAL-01) MAXIMUM SIZE OF COMPILED CONTRACT EXCEEDED - HIGH

Description:

The Terra blockchain has a default limit of the size of a compiled contract in the form of a WASM file, which is statically set to 600kb.

This value can be checked in the terra-money repository.

During the deployment of the test environment, it was noticed that the Generator contract clearly exceeds the previous limit, eventually compiling to a size of 722kb, which prevents the successful launch of the environment.

Code Location:

Listing 1: Listing all compiled artifacts: (Line 6)

```
artifacts % ls -la
total 10360
drwxr-xr-x  20 jakub  staff     640 May 10 20:33 .
drwxrwxr-x@ 16 jakub  staff     512 May 10 20:33 ..
-rw-r--r--   1 jakub  staff  333351 May 10 20:30 astroport_factory.wasm
-rw-r--r--   1 jakub  staff  723607 May 10 20:30 astroport_generator.wasm
-rw-r--r--   1 jakub  staff  221672 May 10 20:30 astroport_generator_proxy_to_mirror.wasm
-rw-r--r--   1 jakub  staff  391928 May 10 20:31 astroport_maker.wasm
-rw-r--r--   1 jakub  staff  235974 May 10 20:31 astroport_oracle.wasm
-rw-r--r--   1 jakub  staff  433086 May 10 20:31 astroport_pair.wasm
-rw-r--r--   1 jakub  staff  344358 May 10 20:31 astroport_pair_anchor.wasm
-rw-r--r--   1 jakub  staff  460021 May 10 20:31 astroport_pair_stable.wasm
-rw-r--r--   1 jakub  staff  575266 May 10 20:32
```

```
↳ astroport_pair_stable_bluna.wasm
14 -rw-r--r--   1 jakub  staff  295528 May 10 20:32 astroport_router.
↳ wasm
15 -rw-r--r--   1 jakub  staff  228177 May 10 20:32 astroport_staking
↳ .wasm
16 -rw-r--r--   1 jakub  staff  288096 May 10 20:32 astroport_token.
↳ wasm
17 -rw-r--r--   1 jakub  staff  239498 May 10 20:32 astroport_vesting
↳ .wasm
18 -rw-r--r--   1 jakub  staff  179137 May 10 20:33
↳ astroport_whitelist.wasm
19 -rw-r--r--   1 jakub  staff  311094 May 10 20:33
↳ astroport_xastro_token.wasm
20 -rw-r--r--   1 jakub  staff    1371 May 10 20:33 checksums.txt
21 -rw-r--r--   1 jakub  staff    1941 May 10 20:30
↳ checksums_intermediate.txt
```

Listing 2: rpc error when deploying generator:

```
1 data: {
2     error: 'rpc error: code = InvalidArgument desc = failed to
↳ execute message; message index: 0; contract size is too huge:
↳ store wasm contract failed: invalid request'
3 }
```

Risk Level:

Likelihood - 5
Impact - 3

Recommendation:

Due to contract size restrictions, the generator contract size should be reduced by optimizing or by transferring some of its content to other related contracts.

Remediation plan:

SOLVED: The issue was solved during the audit in the following commit:
- 6b72e1b994ab30731e7da78ae5abce23d66f7e58.

3.2 (HAL-02) LACK OF BAN FUNCTIONALITY WITHIN BOOSTS - MEDIUM

Description:

The Generator contract did not appear to implement a method to blacklist or ban addresses that misuse or abuse the ASTRO boost for a given generator.

Code Location:

Listing 3: The function responsible for implementing boosting

```
fn checkpoint_user_boost(
    deps: DepsMut,
    env: Env,
    info: MessageInfo,
    generators: Vec<String>,
    user: Option<String>,
) -> Result<Response, ContractError> {
    let config = CONFIG.load(deps.storage)?;
    let recipient_addr = if let Some(user) = user {
        addr_validate_to_lower(deps.api, &user)?
    } else {
        info.sender
    };

    let mut generator_limit = CHECKPOINT_GENERATORS_LIMIT;
    if let Some(limit) = config.checkpoint_generator_limit {
        generator_limit = limit;
    }

    if generators.len() > generator_limit as usize {
        return Err(ContractError::GeneratorsLimitExceeded {});
    }

    let mut send_rewards_msg: Vec<WasmMsg> = vec![];
    for generator in generators {
        let generator_addr = addr_validate_to_lower(deps.api, &
↳ generator)?;
```

```
27
28          // calculates the emission boost  only for user who has LP
↳  in generator
29          if USER_INFO.has(deps.storage, (&generator_addr, &
↳ recipient_addr)) {
30              let user_info =
31                  USER_INFO.compatible_load(deps.storage, (&
↳ generator_addr, &recipient_addr))?;
32
33              let mut pool = POOL_INFO.load(deps.storage, &
↳ generator_addr)?;
34              accumulate_rewards_per_share(&deps.querier, &env, &
↳ generator_addr, &mut pool, &config)?;
35
36              send_rewards_msg.append(&mut send_pending_rewards(
37                  deps.as_ref(),
38                  &config,
39                  &pool,
40                  &user_info,
41                  &recipient_addr,
42              )?);
43
44              // Update user's amount
45              let amount = user_info.amount;
46              let mut user_info = update_user_balance(user_info, &
↳ pool, amount)?;
47
48              // Update user's virtual amount
49              update_virtual_amount(
50                  deps.as_ref(),
51                  &env,
52                  &config,
53                  &mut pool,
54                  &mut user_info,
55                  &recipient_addr,
56                  &generator_addr,
57              )?;
58
59              USER_INFO.save(deps.storage, (&generator_addr, &
↳ recipient_addr), &user_info)?;
60              POOL_INFO.save(deps.storage, &generator_addr, &pool)?;
61          }
62      }
63
```

```
64      Ok(Response::new()
65          .add_attribute("action", "checkpoint_user_boost")
66          .add_messages(send_rewards_msg))
67  }
68
```

Risk Level:

Likelihood - 3
Impact - 3

Recommendation:

A functionality should be introduced that allows for blacklisting, the banning of addresses that misuse the boost mechanism for a given generator.

Remediation plan:

RISK ACCEPTED: The Astroport team presented the use of the checkpoint_user_boost function as an activity to mitigate the risk of the issue.

3.3 (HAL-03) OVERFLOW CHECKS NOT SET FOR 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 each individual contract and package, which could have unexpected consequences if the project is refactored.

Code Location:

Listing 4: Resources affected

```
contracts/factory/Cargo.toml
contracts/tokenomics/generator/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:

ACKNOWLEGED: The Astroport team acknowleged the risk of this finding.

THANK YOU FOR CHOOSING

// HALBORN