SecurityDAO

CW-Unity-Prop Audit

CW Unity Proposal CosmWasm Contract Audit

Prepared by: Logan Cerkovnik
Date: April 3rd, 2022

TVL : **$>=120M USD equivalent**

... + more about threat model

Outline

Document Revision History

Version	Modification	Date	Author
0.2	**Created**	**3/29/2022**	**Logan Cerkovnik**
0.3	**Update for Sudo Entrypoint**	**4/3/2022**	**Logan Cerkovnik**
0.4	**Update for feedback**	**4/4/2022**	**Logan Cerkovnik**

Contact

Contact	Organization	Email
Logan Cerkovnik	Security DAO	logan@secdao.xyz
Paul Wagner	Security DAO	paul@secdao.xyz
Barton Rhodes	Security DAO, DAO DAO	barton@secdao.xyz

Executive Overview

Audit Summary

Security Dao worked on an engagement with the Juno Community and Juno Core Dev group from 3/25/2022 through 4/03/2022 to conduct a security assessment of the unity proposal contracts to ensure security, trust, and communication for prop 18.

The security engineers involved with the audit are security and blockchain smart contract security experts with advanced knowledge of smart contract exploits.

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, Security Dao did not identify any impactful improvements or vulnerabilities for the cw-unity-prop contract.

The one minor improvement that could be taken is to fix the dependencies used in cosmwasm-std v1.0.0beta7 to not use yanked dependencies.

External threats such as intercontract functions and calls should be validated for expected logic and state and are not covered within the scope of this audit. Only direct rpc contract interaction is considered here and any UI components or frontend wasm interactions are excluded from the audit scope. The cosmwasm-std v1.0.0beta7 sudo entrypoint feature is also considered out of scope for purposes of this audit. This audit will not cover the underlying code in either junod or wasmd that implements the sudo entry point.

Test Approach and Methodology

Security DAO performed a combination of manual review of the code and automated security testing.

The following phases were used throughout the audit:

- Research into the architecture, purpose, and use of the platform

- Manual code review and walkthrough
- Manual Assessment of the use and safety for critical rust variables and functions in scope to identify any contracts logic related vulnerability
- Fuzz Testing (securitydao fuzzing tool)
- Check Test Coverage (cargo tarpaulin)

Contract	% coverage	Lines Covered
cw-unity-prop	99.35%	457/460

- Scanning of Rust files for vulnerabilities (cargo audit)

Contract	Dependency	Version	Warning
cw-unity-prop	const-oid	0.6.0	yanked
cw-unity-prop	crypto-bigint	0.2.2	yanked

Risk Methodology

Risk Likelihood and impact scales 1 through 5 where 5 is the most severe

Risk Likelihood Scale

1 low	2 unlikely	3 possible	4 likely to happen	5 high

`least severe` `most severe`

A low likelihood risk indicates that the likelihood of attack is low because of obscurity or requiring additional exploits to utilize, a possible attack is one that is possible but not an attack method commonly seen in the wild or well-known, and high risk likelihood represents an exploit extremely likely to be used, readily apparent, or commonly been used in the past against similar systems

Risk Impact Scale

1 low	2 limited	3 Impactful	4 critical	5 severe

`least severe` `most severe`

In the context of smart contracts, a low risk impact might be something associated with limited scope or a preventive best practice, an impactful risk may result in large loss of funds but not in a systematic way, and a severe risk impact could result in substantial loss of funds in a systematic way.

Scope

The primary target for the audit is cw-unity-prop cosmwasm smart contract crate. User interface and cross contract messages are considered out of scope for this work.

- **Repo:** `Private:` https://github.com/CosmosContracts/cw-unity-prop
- **Commit hash: 54c1fb6fa5157bcb76b8e9b9736acc6c59a99575**

Action Plan

This audit primarily focuses on the smart contract code and functionality. However, it is not possible to test the sudo entrypoint outside of the cw-multitest tooling without a simulated governance environment in either Uni-2 testnet or a single local validator setup with a new L1 chain. It is the auditor's understanding that these tests will be completed and this audit will be updated with the results from those tests.

Low Level of Effort to Fix and Low Impact

- **(SEC - 13)** Upgrade Yanked Dependencies

Testing Functionality of Governance Proposal

- Local Testing of Single Validator Set Chain
- Uni-2 Simulated Governance proposal vote and action

Assessment Summary and Findings Overview

Findings and Tech Details

(SEC - 13) Yanked Dependencies

Severity Low / Impact Low

Description

Dependencies for const-oid version 0.6.0 and crypto-bigint 0.2.2 are both yanked. This is a cosmwasm-std 1.0.0beta7 dependency issue and the cosmwasm-std maintainers should fix the yanked crates eventually.

Code Location

https://github.com/CosmosContracts/cw-unity-prop/blob/main/Cargo.lock

Risk Level

The risk likelihood is low and the impact is low

Recommendation

The maintainers of cosmwasm-std need to remove yanked crates from the next version of cosmwasm-std crate.

Remediation Plan

None, developers accept the risk of using yanked crates used by maintainers of the cosmwasm-std.

Findings on Sudo Entrypoint Development

SecurityDAO usually focuses only on vulnerabilities and functionality for provided contracts. We felt some commentary on the sudo entrypoint that became available after prop 17 would be helpful because of its use in the prop 18 contract.

We felt the following analysis would be helpful for the Juno community and all parties to Juno prop 18:

- sudo entrypoint has been used in cosmwasm smart contracts prior to prop 18
- sudo entrypoint feature development began prior to Juno prop 4 and 16
- sudo entrypoint feature was developed independently of Juno devs and Juno Core Dev team

Prior Uses of The Sudo Entry point

The first uses of the sudo entry point in a cosmwasm contract that can be found in the hackathon contracts in the cosmwasm repo and the poe contracts from the tg4-engagement contract from Confio. These contracts have been under development since at least January 2022 and somewhat stable since Feb 21st 2022.

https://github.com/confio/poe-contracts/blob/1d3d49f07a9bdc1ba6e3121fb5801fa6ef477ca2/contracts/tg4-engagement/src/contract.rs#L618

https://github.com/CosmWasm/cosmwasm/blob/main/contracts/hackatom/src/contract.rs#L52

We find it encouraging that there is at least one prior open source example using this cosmwasm functionality prior to Juno prop 16/17/18.

Audit History of the Sudo Entry Point

We are not able to find an openly available security audit covering the sudo entrypoint at this time. However this does not mean that this feature has not been covered under past audits. We are reaching out to Confio to determine the audit status of the cosmwasm sudo entry point.

Development History of the Sudo Entry Point

We have reviewed the development history related to the sudo entry point. We have found that historical development of the sudo or system functionality precedes not only Juno prop 16/17/18 but also preceded Juno prop4. Confio developers had worked on this feature as early as May 2021 and are independent of the Juno ecosystem / developers.
Juno devs enabled this feature by upgrading the external wasmd and cosmswasm-std dependencies for Junod. We believe that there is a low probability of this feature not being integrated into Juno correctly. We have not formally audited either the wasmd or Juno codebase for security purposes here.

The following development history, documentation, and changelogs were reviewed here:

Cosmwasm docs:

- https://docs.cosmwasm.com/docs/1.0/
- https://docs.cosmwasm.com/docs/1.0/smart-contracts/sudo
- https://docs.cosmwasm.com/docs/1.0/smart-contracts/entry-points

Cosmwasm-std:

- https://github.com/CosmWasm/cosmwasm/tree/v1.0.0-beta7/packages/std
- https://github.com/CosmWasm/cosmwasm/tree/v1.0.0-beta7/packages/vm
- https://github.com/CosmWasm/cosmwasm/blob/v1.0.0-beta7/packages/std/src/exports.rs#L162

Cosmwasm-std docs.rs:

- https://docs.rs/cosmwasm-std/1.0.0-beta7/cosmwasm_std/index.html

Juno:

- https://github.com/CosmosContracts/juno/releases/tag/v2.3.0
- Upgrade mainline wasmd version to 0.24.0 by @the-frey in #158

Juno docs:

- https://docs.junonetwork.io/juno/readme

Wasmd:

- https://github.com/CosmWasm/wasmd/tree/v0.24.0
- https://github.com/CosmWasm/wasmd/blob/3bc0bdeab3fa2b3f7de745622226ff36c2ec6d6a/CHANGELOG.md
- It would appear from changelog that sudomsg protobuf / event first appears in wasmd v0.22.0 in Add governance proposals for Wasm Execute and Sudo pr #730 (ethanfrey)

Cw-plus/cw-multitest:

- https://github.com/CosmWasm/cw-plus/tree/v0.13.0/packages/multi-test
- https://github.com/CosmWasm/cw-plus/blob/ce603a49f3d0e222a707711cba3915e0e27de434/CHANGELOG.md
- Expose sudo powers on Router we give to Modules #453 (ethanfrey)