OBJECT COMPUTING

Object Computing, Inc.
12140 Woodcrest Executive Dr.
Suite 300
St. Louis, MO 63141
+1 314.579.0066

Alien Worlds referendum Contract Audit (EOS)

Source for this analysis is github: Alien-Worlds/contracts/referendum
Commit tag: 722e38267153bb53b37431aa86bb0959bd5485e7
Forked github: eosdac/eosdac-contracts
Forked Commit tag: f213b0f07c425912692c38e8b7e07ad99da9ddca
Files: contracts/referendum/referendum.[ch]pp
Tests: contracts/referendum/referendum.tests.ts
Ricardian contracts: contracts/referendum/referendum.contracts.md
Business requirements and other engineering artifacts:
https://eosdac.io/tools/smart-contracts-explained/
contracts/daccustodian/README.md

Executive Summary

No critical security violations were detected in this audit.

Remark	Minor	Major	Critical
0	0	0	0

Participants

Primary auditor is Jack DiSalvatore and Phil Mesnier reviewed.

Tools

Static analyzers

- EOSafe appears to be an Academic exercise last updated nearly 4 years ago
- EOSlime is a javascript tool that is also at least 2 years old
- Klevoya is a wasm analyzer tool using simple pattern matches

No static analyzers were used in this exercise. I tried to build the EOSafe tool, but it would not compile. The EOSlime tool was too complicated for me to determine how to build it. Klevoya was not useful because I could not generate an ABI file for the mining contract.

Known Vulnerabilities

List origin github:slowmisg/**eos-smart-contract-security-best-practices**/Readme_EN.md commit tag 5f77e19e50373d341e17a003c492388e9891a2c0

- **Numerical Overflow:** when doing token math, use the `asset` object operands and do not perform operations on the `uint256` that comes from `asset.amount`
- **Authorization Check:** make sure function parameters are consistent with the caller and use `require_auth ` to check
- **Apply Check:** if the contract implements an `apply` function, bind each key action and code to meet the requirements, in order to avoid abnormal and illegal calls.
- **Transfer Error Prompt:** When processing a notification triggered by `require_recipient`, ensure that `transfer.to` is `_self`
- **Random Number Practice:** Random number generator algorithm should not introduce controllable or predictable seeds

We relied on visual inspection to look for instances of code that were similar to the examples found in the above document as well as best practices accumulated through many years of experience.

Findings And Recommendations

We examined 6 tables, 10 actions and 5 helper functions.

Remarks
- None

Minor
- None

Major
- None

Critical
- None

Table/Singleton	Notes
using configscontainer = eosio::singleton<"config2"_n, contr_config>;	No issues.
using candperms_table = multi_index<"candperms"_n, candperm>;	No issues.
using config_container = eosio::singleton<"config"_n, config_item>;	No issues.
using referenda_table = eosio::multi_index<"referendums"_n, referendum_data, indexed_by<"byproposer"_n, const_mem_fun<referendum_data, uint64_t, &referendum_data::by_proposer>>>;	No issues.
using votes_table = eosio::multi_index<"votes"_n, vote_info>;	No issues.
using deposits_table = eosio::multi_index<"deposits"_n, deposit_info, indexed_by<"bysym"_n, const_mem_fun<deposit_info, uint128_t, &deposit_info::by_sym>>>;	No issues.

Action	Notes	Ricardian Contract
ACTION updateconfig(config_item config, name dac_id);	No issues.	Yes
ACTION propose(name proposer, name referendum_id, uint8_t type, uint8_t voting_type, string title, string content, name dac_id, vector<action> acts);	No issues.	Yes
ACTION cancel(name referendum_id, name dac_id);	No delta.	Yes
ACTION vote(name voter, name referendum_id, uint8_t vote, name dac_id);	No delta.	Yes
ACTION exec(name referendum_id, name dac_id);	No delta.	Yes
ACTION clean(name account, name dac_id);	No delta.	No
ACTION refund(name account);	No delta.	Yes
ACTION clearconfig(name dac_id);	No issues.	No
ACTION stakeobsv(vector<account_stake_delta> stake_deltas, name dac_id);	No delta.	No
[[eosio::on_notify("*::transfer")]] void receive(name from, name to, asset quantity, string memo);	No delta.	No

Helper Functions	Notes
bool _check_transaction_authorization(const char *trx_data, uint32_t trx_size, const char *pubkeys_data, uint32_t pubkeys_size, const char *perms_data, uint32_t perms_size)	No delta.

bool hasAuth(vector<action> acts);	No delta.
uint8_t calculateStatus(name referendum_id, name dac_id);	No issues.
void proposeMsig(referendum_data ref, name dac_id);	No issues.
void checkDAC(name dac_id);	No delta.

Test Case	Notes
updateconfig	No issues.
propose	No issues.
vote	No issues.
exec	No issues.
msig exec	No issues.

Test Coverage

The amount of actions tested divided by the number of total actions.

4 / 10 = 40%

No methodology definitively proves the absence of vulnerabilities. Following assessment and remediation, modifications to an application, its platform, network environment, and new threat vectors may result in new application security vulnerabilities.