OBJECT COMPUTING

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

Alien Worlds Shining Contract Audit (EOS)

Source for this analysis is github: Alien-Worlds/alienworlds-contracts
Commit tag 1a656355b7c2c217898bde31c1929ab7029fe3b6
Files contracts/eosio.token/eosio.token.[ch]pp
No tests available
No ricardian contract sources were available
Business requirements and other engineering artifacts were found in the “ClickUp” document found at https://doc.clickup.com/d/h/hfd6b-9788/49593a316fe4392/hfd6b-1448

Executive Summary

No obvious signs of severe or critical security violations were detected in this audit.

Lack of ricardian contracts is not necessarily a risk but is a lost opportunity for ensuring that the coded logic is congruent with the identified business needs.

Participants

Primary Auditors were Jack DiSalvatore, with Ciju John providing a review.

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
- **Rollback Attack:** There is a time difference between when an API node initiates the request and synchronizes to the BP node. (In the context of a gambling dapp) Attackers use this vulnerability to place bets on the API node. If they find that there is no winning, they can return the EOS for each bet so that they can rejoin the game without cost. Attackers can repeat this process until they have a winning bet. One solution to this is for your dapp to use read/write separation and have Read-Only and Write-Only nodes to interact with the smart contract.

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 5 contract actions. The contract action is documented in clickup, but not the modifications to the private function.

-

Table Signatures	In ClickUp	Notes
using lookups_table = multi_index<"lookups"_n, lookups>;	No	Minor Line#17 primary key `from` should be a `uint64_t`.
using deposits_table = multi_index<"deposits"_n, deposits>;	No	No issue.
using shines_table = multi_index<"shines"_n, shines>;	No	No issue.
using config_table = eosio::singleton<"config"_n, config_item> ;	No	No issue.

Action Signatures	In ClickUp	Notes
ACTION addlookup(uint32_t from, uint32_t to, asset cost, uint8_t qty, time_point_sec start_time, bool active);	Yes	No issue.
ACTION setgenesisid(uint64_t genesis_id);	No	No issue.

ACTION clearlookups();	No	No issue.
[[eosio::on_notify(NFT_CONTRACT_STR "::transfer")]] ACTION nfttransfer(name from, name to, vector<uint64_t> asset_ids, string memo);	Yes	Remark Line#57 `template_check` does not need to be redefined, just use `template_id` instead. Minor Line#64 check the length of asset_ids before entering the for-loop to avoid exceeding the inline action call stack depth. (max is 10 on the EOS mainnet) Remark Line#85 & 88 compare the entire asset and not just the amount. i.e. `existing_deposit->quantity == lookup->cost` Remark Line#115-120 remove commented out burn code and update the documentation.
[[eosio::on_notify(TOKEN_CONTRACT_STR "::transfer")]] ACTION tlmtransfer(name from, name to, asset quantity, string memo);	Yes	Minor Add a check to make sure the asset being deposited is the TLM token.

Setup Script	**In ClickUp**	**Notes**
get_asset_ids.js		No issues.
populate_shining.js		No issues.
set_genesis_id.js		No issues.
transact_cleos.js		No issues.
update_prices.js		Remark Line#21-25 `sleep` function is unused and can be removed. Remark Line#58 & 89 replace infinite while loop with a boolean variable, and toggle it true/false to exit the loop instead of using `break` statements.
validate_shining.js		No issues.

Summary

Remark	Minor	Major	Critical
5	3	0	0

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.