OBJECT COMPUTING

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

Alien Worlds Shared Headers Audit (EOS)

Source for this analysis is github: Alien-Worlds/contract-shared-headers/
Commit tag: 722e38267153bb53b37431aa86bb0959bd5485e7

Files: contract-shared-headers/common_utilities.hpp
contract-shared-headers/config.hpp
contract-shared-headers/daccustodian_shared.hpp
contract-shared-headers/dacdirectory_shared.hpp
contract-shared-headers/eosdactokens_shared.hpp
Tests: N/A
Ricardian contracts: N/A
Business requirements and other engineering artifacts: N/A

Executive Summary

No critical security violations were detected in this audit.

Remark	Minor	Major	Critical
2	0	0	0

Participants

Primary auditors are Jack DiSalvatore and Phil Mesnier, 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

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 8 tables, 15 actions and 6 helper functions.

Remarks

- Mixing snake-case and camel case for function names.
- Dacdirectory_shared.hpp Line#94 replace “{ }” with “( )” to be consistent with other declarations.

Minor

-

Major

-

Critical

-

common_utilities.hpp

Functions	**Notes**

uint128_t combine_ids(const uint8_t &boolvalue, const uint64_t &longValue)	No delta.
uint128_t combine_ids(const uint16_t &value, const uint64_t &longValue)	No delta.
static const uint128_t combine_ids(const uint64_t &x, const uint64_t &y)	No delta.
static const checksum256 combine_ids(const uint64_t &w, const uint64_t &x, const uint64_t &y, const uint64_t &z)	No delta.
static const __uint128_t raw_from_extended_symbol(const extended_symbol &symbol)	No delta.
template <typename T> inline std::string toString(const T &x)	No issues.
template <typename... Args> inline char *fmt(const std::string_view format, Args const &...args)	No issues.
template <typename... Args> inline void check(bool pred, const std::string_view format, Args const &...args)	No issues.
template <typename Table, typename Pk, typename Function> inline bool upsert(Table &table, const Pk &pk, const eosio::name payer, const Function &updater)	No issues.

config.hpp

Functions	**Notes**
N/A	No functions to review.

daccustodian_shared.hpp

Tables/Singletons	**Notes**
custodians	No issues.
candidates	No issues.
weights	No issues.

dacdirectory_shared.hpp

Tables/Singletons	**Notes**
dacs	No issues.
nftcache	No issues.

Functions	Notes
const dac dac_for_id(eosio::name id)	No issues.
const dac dac_for_symbol(eosio::extended_symbol sym)	No issues.
const std::optional<dac> dac_for_owner(eosio::name owner)	**Remark**: Line#94 replace “{ }” with “( )” to be consistent with other declarations.

eosdactokens_shared.hpp

Variable Definitions	Notes
account_stake_delta	No delta.
account_balance_delta	No delta.
account_weight_delta	No delta.

Tables/Singletons	Notes
memberterms	No issues.
stat	No issues.
members	No issues.
accounts	No issues.
stakes	No issues.
unstakes	No issues.
staketime	No issues.

Functions	Notes
asset get_supply(name code, symbol_code sym)	No issues.
asset get_balance(name owner, name code, symbol_code sym)	No issues.
asset get_balance_graceful(name owner, name code, symbol sym)	No issues.
asset get_liquid(name owner, name code, symbol sym)	No issues.
asset get_staked(name owner, name code, symbol sym)	No delta.
static void assertValidMember(eosio::name member, eosio::name dac_id)	No issues.

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.