Audit Report

DAO DAO Updates

v1.0

October 12, 2023

Table of Contents

Table of Contents **2**
License **4**
Disclaimer **4**
Introduction **6**
Purpose of This Report 6
Codebase Submitted for the Audit 6
Methodology 9
Functionality Overview 9
How to Read This Report **10**
Code Quality Criteria 11
Summary of Findings **12**
Detailed Findings **14**
1. Hooks are not dispatched during stake and unstake messages 14
2. DAO is unable to control the cw-tokenfactory-issuer contract for existing factory tokens 14
3. AbsoluteCount can be configured to be greater than the total NFT supply 15
4. AbsoluteCount threshold for a new token is not validated 15
5. Protocol addresses can be blacklisted 16
6. Group contract attribute key emits as address string 16
7. Newly instantiated NFT contract can have zero NFTs, locking the DAO 16
8. Adding too many hooks may cause stake and unstake messages to fail 17
9. Incorrect events are emitted 17
10. Lack of denom validation 18
11. SG721 NFT creator role is not set to the DAO address 18
12. BlockBeforeSend hook executes upon minting or burning 19
13. Unstaking duration is not validated in the dao-voting-cw721-staked contract 19
14. Absolute count is not validated when instantiating NFT contract through factory contract 20
15. DAO will be locked if the minter is set to the DAO address 20
16. DAO will not accept pending ownership transfer automatically 21
17. TokenContract query will fail for TokenInfo::Existing tokens 21
18. BEFORE_SEND_HOOK_FEATURES_ENABLED is not exposed through smart queries 22
19. Misleading from attribute when burning funds 22
20. Code reusability can be improved 23
21. Inconsistent attribute names and orders 23
22. Incorrectly spelled storage constant name 24
23. Counterintuitive variable names 24
24. Gas consumption can be reduced by setting contract admin to DAO 25

25. Unnecessary reply_always when instantiating new token 25
26. Centralization risk on token management 26
27. Misleading comments 26
28. “Migrate only if newer” pattern is not followed 27
29. Contracts should implement a two step ownership transfer 27
30. Unused function in codebase 28
31. Freezing mechanism includes a bypass exception 28

License

Disclaimer

This audit has been performed by

Oak Security

https://oaksecurity.io/
info@oaksecurity.io

Introduction

Purpose of This Report

Oak Security has been engaged by Osmosis Grants Company to perform a security audit of updates to the DAO DAO smart contracts.

The objectives of the audit are as follows:

1. Determine the correct functioning of the protocol, in accordance with the project specification.
2. Determine possible vulnerabilities, which could be exploited by an attacker.
3. Determine smart contract bugs, which might lead to unexpected behavior.
4. Analyze whether best practices have been applied during development.
5. Make recommendations to improve code safety and readability.

This report represents a summary of the findings.

As with any code audit, there is a limit to which vulnerabilities can be found, and unexpected execution paths may still be possible. The author of this report does not guarantee complete coverage (see disclaimer).

Codebase Submitted for the Audit

The audit has been performed on the following target:

Repository	https://github.com/DA0-DA0/dao-contracts
Commit	`7f89ad1604e8022f202aef729853b0c8c7196988`
Scope	The scope was restricted to changes since our last audits of DAO DAO, including a refactoring for easier publication as crates, excluding changes to the following contracts that we have not audited in the past: - `contracts/external/cw-fund-distributor` - `contracts/external/dao-migrator` Our last audits were performed on commit `0b5cae57fecbbadb1045f3dc2bb4ad4fe5a98ee8` for the following paths: - `contracts/external/cw-vesting`

	- `contracts/external/cw-payroll-factory` - `packages/cw-wormhole` On commit `74bd3881fdd86829e5e8b132b9952dd64f2d0737` for the following paths: - `contracts/dao-core` - `contracts/external/cw-admin-factory` - `contracts/external/cw-token-swap` - `contracts/pre-propose/dao-pre-propose-approval-single` - `contracts/pre-propose/dao-pre-propose-approver` - `contracts/pre-propose/dao-pre-propose-multiple` - `contracts/pre-propose/dao-pre-propose-single` - `contracts/proposal/dao-proposal-multiple` - `contracts/proposal/dao-proposal-single` - `contracts/staking/cw20-stake` - `contracts/staking/cw20-stake-external-rewards` - `contracts/staking/cw20-stake-reward-distributor` - `contracts/voting/dao-voting-cw4` - `contracts/voting/dao-voting-cw20-staked` - `contracts/voting/dao-voting-cw721-staked` - `contracts/voting/dao-voting-native-staked` - `contracts/voting/dao-voting-staking-denom-staked` - Relevant files in the `packages/*` folder And on commit `490a9e8eb1704d0207d03286d065693b9e17fa85` for the following path: - `contracts/proposal/dao-proposal-condorcet`
Fixes verified at commit	`3518a283d1951c42fb71328556fbfb039fd2b210` Note that changes to the codebase beyond fixes after the initial audit have not been in the scope of our fixes review.

Repository	https://github.com/DA0-DA0/dao-contracts
Commit	`37125086a464050af62bb4d15d936653cae61f31`
Scope	The scope was limited to the following new contracts: - `contracts/external/cw-tokenfactory-issuer` - `contracts/voting/dao-voting-token-factory-staked` And to changes to the following contracts since the updates and refactoring mentioned above, which was reviewed at commit `7f89ad1604e8022f202aef729853b0c8c7196988`: - `contracts/voting/dao-voting-cw4` - `contracts/voting/dao-voting-cw721-staked` - `contracts/voting/dao-voting-native-staked`

Fixes verified at commit	`3518a283d1951c42fb71328556fbfb039fd2b210` Note that changes to the codebase beyond fixes after the initial audit have not been in the scope of our fixes review.

Repository	https://github.com/DA0-DA0/dao-contracts
Commit	`f01cc597c1ea8670096832d77e548a44062badd2`
Scope	This audit covers the factory pattern and remaining fixes implemented in pull request 750, which includes the commits from `5a85380f7ace0d1f1f3bfa06d01430f4f44de865` to `f01cc597c1ea8670096832d77e548a44062badd2`. The factory pattern is intended to support the Fairburn mechanism for Stargaze NFTs.
Identifier	In this report, all paths pointing to this pull request are prefixed with `factory-nft:`
Fixes verified at commit	`ac1667cc1582ec5d391a8b2cfa5e88dee97758bf` Note that changes to the codebase beyond fixes after the initial audit have not been in the scope of our fixes review.

Methodology

The audit has been performed in the following steps:

1. Gaining an understanding of the code base's intended purpose by reading the available documentation.
2. Automated source code and dependency analysis.
3. Manual line-by-line analysis of the source code for security vulnerabilities and use of best practice guidelines, including but not limited to:
    a. Race condition analysis
    b. Under-/overflow issues
    c. Key management vulnerabilities
4. Report preparation

Functionality Overview

The submitted code implements the smart contracts for DAO DAO, which creates a modular framework for creating DAOs, including staking and voting functionalities.

How to Read This Report

This report classifies the issues found into the following severity categories:

Severity	Description
Critical	A serious and exploitable vulnerability that can lead to loss of funds, unrecoverable locked funds, or catastrophic denial of service.
Major	A vulnerability or bug that can affect the correct functioning of the system, lead to incorrect states or denial of service.
Minor	A violation of common best practices or incorrect usage of primitives, which may not currently have a major impact on security, but may do so in the future or introduce inefficiencies.
Informational	Comments and recommendations of design decisions or potential optimizations, that are not relevant to security. Their application may improve aspects, such as user experience or readability, but is not strictly necessary. This category may also include opinionated recommendations that the project team might not share.

The status of an issue can be one of the following: **Pending,** **Acknowledged**, or **Resolved**.

Note that audits are an important step to improving the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of the system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**. We include a table with these criteria below.

Note that high complexity or low test coverage does not necessarily equate to a higher risk, although certain bugs are more easily detected in unit testing than in a security audit and vice versa.

Code Quality Criteria

The auditor team assesses the codebase's code quality criteria as follows:

Criteria	Status	Comment
Code complexity	**Low-Medium**	-
Code readability and clarity	**Medium-High**	Most functions are well-documented with concise and clear code comments.
Level of documentation	**High**	Detailed documentation was available at https://github.com/DA0-DA0/dao-contracts/wiki/DAO-DAO-Contracts-Design and in `README` files.
Test coverage	**Medium-High**	-

Summary of Findings

No	Description	Severity	Status
1	Hooks are not dispatched during stake and unstake messages	Major	Resolved
2	DAO is unable to control the `cw-tokenfactory-issuer` contract for existing factory tokens	Major	Resolved
3	`AbsoluteCount` can be configured to be greater than the total NFT supply	Minor	Resolved
4	`AbsoluteCount` threshold for a new token is not validated	Minor	Resolved
5	Protocol addresses can be blacklisted	Minor	Resolved
6	Group contract attribute key emits as address string	Minor	Resolved
7	Newly instantiated NFT contract can have zero NFTs, locking the DAO	Minor	Resolved
8	Adding too many hooks may cause stake and unstake messages to fail	Minor	Acknowledged
9	Incorrect events are emitted	Minor	Resolved
10	Lack of denom validation	Minor	Resolved
11	SG721 NFT creator role is not set to the DAO address	Minor	Resolved
12	`BlockBeforeSend` hook executes upon minting or burning	Minor	Acknowledged
13	Unstaking duration is not validated in the `dao-voting-cw721-staked` contract	Minor	Resolved
14	Absolute count is not validated when instantiating NFT contract through factory contract	Minor	Resolved
15	DAO will be locked if the minter is set to the DAO address	Minor	Resolved
16	DAO will not accept pending ownership transfer automatically	Informational	Resolved

17	`TokenContract` query will fail for `TokenInfo::Existing` tokens	Informational	Resolved
18	`BEFORE_SEND_HOOK_FEATURES_ENABLED` is not exposed through smart queries	Informational	Resolved
19	Misleading `from` attribute when burning funds	Informational	Resolved
20	Code reusability can be improved	Informational	Resolved
21	Inconsistent attribute names and orders	Informational	Resolved
22	Incorrectly spelled storage constant name	Informational	Resolved
23	Counterintuitive variable names	Informational	Resolved
24	Gas consumption can be reduced by setting contract admin to DAO	Informational	Resolved
25	Unnecessary `reply_always` when instantiating new token	Informational	Resolved
26	Centralization risk on token management	Informational	Acknowledged
27	Misleading comments	Informational	Resolved
28	“Migrate only if newer” pattern is not followed	Informational	Resolved
29	Contracts should implement a two step ownership transfer	Informational	Resolved
30	Unused function in codebase	Informational	Resolved
31	Freezing mechanism includes a bypass exception	Informational	Acknowledged

Detailed Findings

1. Hooks are not dispatched during stake and unstake messages

Severity: Major

In `contracts/voting/dao-voting-native-staked/src/contract.rs:116-117`, `AddHook` and `RemoveHook` messages allow the DAO to add or remove hooks to contracts that will be executed during stake and unstake messages. However, hook messages are not executed when `execute_stake` and `execute_unstake` are executed. For comparison, the `dao-voting-token-factory-staked` contract dispatches hook messages accordingly, as seen in `contracts/voting/dao-voting-token-factory-staked/src/contract.rs:200` and `240`. Consequently, hooks that the DAO adds will not be dispatched when a user stakes or unstakes funds.

Recommendation

We recommend dispatching hook messages in the `execute_stake` and `execute_unstake` functions.

Status: Resolved

2. DAO is unable to control the `cw-tokenfactory-issuer` contract for existing factory tokens

Severity: Major

In `contracts/voting/dao-voting-token-factory-staked/src/contract.rs:582-588`, the `ChangeContractOwner` message is not dispatched to the `cw-tokenfactory-issuer` contract after instantiating it with `TokenInfo::Existing`. This step is needed because the current contract owner is the `dao-voting-token-factory-staked` contract address, not the DAO address.

For comparison, new tokens created with `TokenInfo::New` transfer the ownership to the DAO, as seen in lines `693-700`. Consequently, the DAO cannot execute privileged functions in the `cw-tokenfactory-issuer` contract, preventing the protocol from working as intended.

Recommendation

We recommend dispatching a `ChangeContractOwner` message in the reply handler for `TokenInfo::Existing` so the DAO is able to control the `cw-tokenfactory-issuer` contract.

Status: Resolved

3. `AbsoluteCount` can be configured to be greater than the total NFT supply

Severity: Minor

The `execute_update_active_threshold` function in `contracts/voting/dao-voting-cw721-staked/src/contract.rs:424` does not validate that the `AbsoluteCount` active threshold set is less than or equal to the total NFT count before setting the value. This could allow for a situation where the count is set to a value that exceeds the total number of tokens in existence and thus can never be met.

Recommendation

We recommend performing a `NumTokens` query to the NFT contract to determine the total number of tokens and using that value as the maximum absolute threshold that can be set.

Status: Resolved

4. `AbsoluteCount` threshold for a new token is not validated

Severity: Minor

The `instantiate` function in `contracts/voting/dao-voting-token-factory-staked/src/contract.rs:64` does not explicitly handle the case where the `active_threshold` is specified with an `AbsoluteCount` type. Specifically, it does not ensure the provided count value is larger than zero.

Consequently, the `IsActive` query will always return `true` when the `AbsoluteCount` value is set to zero, allowing `dao-proposal-multiple` and `dao-proposal-single` contracts to create proposals on inactive DAOs.

Recommendation

We recommend explicitly handling the case by validating the parameter and returning an error if the validation fails.

Status: Resolved

5. Protocol addresses can be blacklisted

Severity: Minor

The `blacklist` function in `contracts/external/cw-tokenfactory-issuer/src/execute.rs:381` does not prevent the issuer contract from being blacklisted by any address with a blacklister allowance. This can cause a number of key contract functions to be blocked as the protocol will no longer be able to interact with the `tokenfactory` denom due to the validation in `contracts/external/cw-tokenfactory-issuer/src/hooks.rs:17-18`.

Note that this issue can only arise if a blacklister is compromised or misbehaves. Even if this issue occurs, the contract owner can recover from it by removing the address from the blacklist.

Recommendation

We recommend preventing the issuer contract from being added to the blacklist.

Status: Resolved

6. Group contract attribute key emits as address string

Severity: Minor

In `contracts/voting/dao-voting-cw4/src/contract.rs:102`, the `group_contract` attribute key emits the value as the `address` string. This is incorrect because it should emit the actual contract address from the `address` variable value.

Consequently, event indexers will be unable to index the instantiated group contract address.

Recommendation

We recommend emitting the `group_contract` attribute with the actual contract address value.

Status: Resolved

7. Newly instantiated NFT contract can have zero NFTs, locking the DAO

Severity: Minor

In `contracts/voting/dao-voting-cw721-staked/src/contract.rs:646-655`, the `initial_nfts` sub-messages provided by the contract instantiator will be dispatched to the NFT contract to mint NFTs. Having at least one NFT minted is important to ensure the

DAO can be controlled by the NFT owners. If no NFTs are minted, no one can control the DAO to mint new NFTs, thereby locking the DAO.

The issue is that no validation ensures the `initial_nfts` sub-messages actually mint the NFT. For instance, it is possible to have two messages that first mint the NFT and later burn it. For `sg721` NFTs, the provided sub-messages can also be used to update the collection information instead of minting new NFTs.

Recommendation

We recommend performing a `NumTokens` query message to ensure the returned value matches the length of `initial_nfts` after dispatching the provided sub-messages. Alternatively, verifying that the total number of NFTs is larger than zero will be sufficient if the main concern is to prevent the DAO from being locked.

Status: Resolved

8. Adding too many hooks may cause stake and unstake messages to fail

Severity: Minor

In `contracts/voting/dao-voting-cw721-staked/src/contract.rs:213` and `264`, the `stake_hook_msgs` and `unstake_hook_msgs` functions dispatch all hooks to the configured hook contracts when a user stakes or unstake funds. If there are too many hooks configured, the transaction will fail due to an out-of-gas error. Consequently, users are unable to stake or unstake funds in the contract, preventing it from working as intended.

We classify this issue as minor because only the contract owner can add hooks, which is a privileged address.

Recommendation

We recommend implementing a maximum limit of hooks that can be configured.

Status: Acknowledged

The client stated that it is difficult to design a good pattern to resolve this issue generically, but they will keep it in mind for the future. Should a DAO add too many stake or unstake hooks, it is still possible to vote to remove them.

9. Incorrect events are emitted

Severity: Minor

In `contracts/external/cw-tokenfactory-issuer/src/execute.rs:239`, the `set_whitelister` function emits the `action` attribute value as `set_blacklister`,

which is incorrect. This issue is also present in `contracts/external/cw-tokenfactory-issuer/src/execute.rs:436`, where the `whitelist` function incorrectly emits the `action` attribute value as `blacklist`.

Consequently, event indexers will incorrectly index the emitted action as the executed message, confusing off-chain listeners.

Recommendation

We recommend updating the `set_whitelister` and `whitelist` functions to emit the correct `action` attribute values.

Status: Resolved

10. Lack of denom validation

Severity: Minor

The `instantiate` function of the `voting-native-staked` contract does not validate the denom stored as part of the configuration in `contracts/voting/dao-voting-native-staked/src/contract.rs:61`.

As this piece of data cannot be updated, a mistake or typo would result in the contract needing to be deployed again.

Recommendation

We recommend performing a `Supply` bank query message to ensure the provided denom is valid.

Status: Resolved

11. SG721 NFT creator role is not set to the DAO address

Severity: Minor

The `instantiate` function of the `dao-voting-cw721-staked` contract modifies the contents of the token instantiate message to set the `minter` to the contract address in `contracts/voting/dao-voting-cw721-staked/src/contract.rs:39-44`. However, for SG721 tokens, the privileged `creator` role that can update collection metadata and royalties is not modified.

Consequently, the DAO cannot update the metadata and royalty configurations of the SG721 NFT token.

This issue was discovered independently by the client while fixing issues.

Recommendation

We recommend setting the `creator` role of the SG721 NFT token to the DAO address.

Status: Resolved

12. `BlockBeforeSend` hook executes upon minting or burning

Severity: Minor

The `BlockBeforeSend` `Sudo` message, defined in `contracts/external/cw-tokenfactory-issuer/src/contract.rs:140`, is executed when sending tokens through the `Bank` module and is used by the `cw-tokenfactory-issuer` contract to enforce freezing and blacklisting of addresses. The current implementation does not consider that this includes the mint operation as it calls the `SendCoinsFromModuleToAccount` function here, which calls `SendCoins` here,

In this case, the impact is limited to the minter not being able to mint if the `cw-tokenfactory-issuer` contract is in a frozen state unless the `tokenfactory` module's address is whitelisted. In addition, this incorrect assumption could lead to vulnerabilities in future versions of the code.

Recommendation

We recommend that the `tokenfactory` module's address should be whitelisted upon contract instantiation.

Status: Acknowledged

The client opened an issue at https://github.com/DA0-DA0/dao-contracts/issues/754 to potentially address this issue in the future.

13. Unstaking duration is not validated in the `dao-voting-cw721-staked` contract

Severity: Minor

In `factory-nft:contracts/voting/dao-voting-cw721-staked/src/contract.rs:71` and `379`, the unstaking duration is not validated to ensure the height and time are not zero. In comparison, the validation is performed for the `dao-voting-token-staked` contract in `factory-nft:contracts/voting/dao-voting-token-staked/src/contract.rs:64` and `286`. Consequently, instant staking and unstaking could be possible due to zero staking duration.

We classify this issue as minor because only the DAO can configure the unstaking duration, which is a privileged account.

Recommendation

We recommend calling the `validate_duration` function in `factory-nft:contracts/voting/dao-voting-cw721-staked/src/contract.rs:71` and `379`.

Status: Resolved

14. Absolute count is not validated when instantiating NFT contract through factory contract

Severity: Minor

In `factory-nft:contracts/test/dao-test-custom-factory/src/contract.rs:343-350`, absolute count validation is not performed after instantiating the NFT contract from the `execute_nft_factory` function. In comparison, this validation is performed when instantiating a `cw_tokenfactory_issuer` contract in lines `268-277` from the `execute_token_factory_factory` function.

Consequently, the absolute count can be set to an invalid value, causing the `IsActive` query to always return `true` when the count is set to zero or return `false` by setting the count value higher than the total NFT supply.

Recommendation

We recommend performing absolute count validation after instantiating the NFT contract through the factory contract.

Status: Resolved

15. DAO will be locked if the minter is set to the DAO address

Severity: Minor

In `factory-nft:contracts/test/dao-test-custom-factory/src/contract.rs:93`, the `execute_nft_factory` function instantiates a new NFT contract without support of automatically minting new NFTs. If `cw721_instantiate_msg.minter` is set to the DAO address, it will cause the DAO to be locked because no one owns the initial NFTs. Since proposals cannot be created, new NFTs cannot be minted.

Recommendation

We recommend allowing NFTs to be dynamically minted to prevent the DAO from being locked.

Status: Resolved

16. DAO will not accept pending ownership transfer automatically

Severity: Informational

In `factory-nft:contracts/voting/dao-voting-cw721-staked/src/contract.rs:685`, the `UpdateOwnership` message is dispatched to transfer the NFT mint ownership to the DAO address. However, the response data is not set with `ModuleInstantiateCallback` to accept the NFT mint ownership automatically in the DAO contract.

Consequently, the DAO members need to create a proposal to accept the mint ownership of the NFT contract manually.

This issue is also present in `factory-nft:contracts/test/dao-test-custom-factory/src/contract.rs:325-334` from the `execute_token_factory_factory` function.

Recommendation

We recommend automatically accepting the mint ownership transfer by setting the response data with `ModuleInstantiateCallback`, similar to `factory-nft:contracts/voting/dao-voting-token-staked/src/contract.rs:701-709`.

Status: Resolved

17. `TokenContract` query will fail for `TokenInfo::Existing` tokens

Severity: Informational

In `factory-nft:contracts/voting/dao-voting-token-staked/src/contract.rs:405`, the `TokenContract` query tries to `load` the `TOKEN_ISSUER_CONTRACT` storage value. Since tokens instantiated as `TokenInfo::Existing` enum will not be stored in the `TOKEN_ISSUER_CONTRACT` storage, the `TokenContract` query will always error for them.

Recommendation

We recommend using the `may_load` function and returning `Option<Addr>` to prevent errors for `TokenInfo::Existing` tokens.

Status: Resolved

18. `BEFORE_SEND_HOOK_FEATURES_ENABLED` is not exposed through smart queries

Severity: Informational

In `contracts/external/cw-tokenfactory-issuer/src/contract.rs:146`, no smart queries expose the `BEFORE_SEND_HOOK_FEATURES_ENABLED` storage state value. This forces users and other contracts to perform a raw query to read the stored value, decreasing user experience.

Recommendation

We recommend exposing a smart query that returns the `BEFORE_SEND_HOOK_FEATURES_ENABLED` storage value.

Status: Resolved

19. Misleading `from` attribute when burning funds

Severity: Informational

In `contracts/external/cw-tokenfactory-issuer/src/execute.rs:121`, the `from` attribute is emitted as the caller's address. This might be misleading because off-chain listeners might expect the `from` address to be the `burn_from_address` address, which is the address that has their funds burnt.

Recommendation

We recommend modifying the `from` attribute name into `burner` and creating a new attribute name called `burn_from_address` to emit the address that has their funds burnt.

Status: Resolved

20. Code reusability can be improved

Severity: Informational

There are a number of validation functions in the scope of this audit that are reimplemented for each contract, decreasing the maintainability and readability of the code.

- The `validate_duraton` validation is duplicated in `dao-voting-native-staked`, `dao-voting-token-factory-staked`, and `cw20-staked`.
- The `active_threshold` validation is duplicated in `contracts/voting/dao-voting-cw20-staked/src/contract.rs:41` and `contracts/voting/dao-voting-token-factory-staked/src/contract.rs:81`.
- The `active_threshold` validation is duplicated in `contracts/voting/dao-voting-cw721-staked/src/contract.rs:89` and `contracts/voting/dao-voting-cw721-staked/src/contract.rs:437`.

Recommendation

We recommend consolidating redundant functions into a shared package.

Status: Resolved

21. Inconsistent attribute names and orders

Severity: Informational

In several instances in the codebase, there are inconsistencies where the same attribute values are emitted with different attribute names or different orderings when emitting events.

- In `contracts/external/cw-tokenfactory-issuer/src/execute.rs:358`, the `allowance` value is emitted as `amount` attribute name. This is inconsistent with the `set_burner` function, which emits the `allowance` attribute name in line `331`.
- In `contracts/voting/dao-voting-cw4/src/contract.rs:199`, new group contracts emits the address with the `group_contract_address` attribute name, while existing group contract emits the address with `group_contract` attribute name in line `102`.
- In `contracts/voting/dao-voting-token-factory-staked/src/contract.rs:704`, new tokens emit the attribute name as `denom` while existing tokens emit it as `token_denom` in line `119`.
- In `contracts/voting/dao-voting-cw721-staked/src/contract.rs:675`,

the `method` attribute is emitted in a later place compared to the `Existing` enum in line `675`, which emits in the first place.

Consequently, event indexers and off-chain listeners might be misled due to different attribute names and attribute orders.

Recommendation

We recommend using the same attribute names and equal orderings for consistency.

Status: Resolved

22. Incorrectly spelled storage constant name

Severity: Informational

The `INITITIAL_NFTS` constant in `contracts/voting/dao-voting-cw721-staked/src/state.rs:23` contains a typo in its name. It is currently spelled as `INITITIAL_NFTS` instead of `INITIAL_NFTS`.

While this does not directly introduce a security risk, it may lead to confusion of future developers and auditors reading the code. Misspelled constants can make the code harder to understand and maintain.

Recommendation

We recommend updating the spelling mentioned above.

Status: Resolved

23. Counterintuitive variable names

Severity: Informational

In `contracts/external/cw-tokenfactory-issuer/src/state.rs:21, 22` and `24`, the `BLACKLISTER_ALLOWANCES`, `WHITELISTER_ALLOWANCES`, and `FREEZER_ALLOWANCES` represent the addresses that can blacklist, whitelist, freeze contract state. The word “allowance” hints that the addresses can only perform actions for a limited amount of time until their allowance is consumed or revoked. However, this is not the case, as the actions can be performed many times without restrictions.

While this does not directly introduce a security risk, potentially misleading variable names can reduce code readability and maintainability.

Recommendation

We recommend modifying the variable names to be more intuitive.

Status: Resolved

24. Gas consumption can be reduced by setting contract admin to DAO

Severity: Informational

In `contracts/voting/dao-voting-cw4/src/contract.rs:69`, the admin of the `cw4_group` contract is set to the current contract address and transferred to the DAO during the reply handler in line `194`. This step is unnecessary and inefficient as it is possible to set the `cw4_group` contract admin to the DAO address directly.

Recommendation

We recommend setting the `cw4_group` contract admin to the DAO address directly to reduce gas consumption and increase code readability.

Status: Resolved

25. Unnecessary `reply_always` when instantiating new token

Severity: Informational

In `contracts/voting/dao-voting-token-factory-staked/src/contract.rs:125`, the instantiation of the `cw-tokenfactory-issuer` contract is dispatched as a `reply_always` sub-message. This sub-message is typically used to handle cases where an error occurred, but the transaction does not need to be reverted.

In this case, there is no need to use the `reply_always` sub-message because no errors need to be handled.

Recommendation

We recommend using `reply_on_success`, similar to line `103`.

Status: Resolved

26. Centralization risk on token management

Severity: Informational

The `force_transfer` function in `contracts/external/cw-tokenfactory-issuer/src/execute.rs:441-469` allows the owner to transfer tokens from one address to another without restrictions.

Although the owner is expected to be a DAO which will reduce centralization, we still consider this function dangerous as it introduces centralization risks, for example tampering with balances that could be used to influence polls.

Recommendation

We recommend removing features that allow privileged addresses to change users' balances freely.

Status: Acknowledged

The client stated that this is an inherited risk stemming from the design of the Token Factory. They have documented how to remove the admin and stated that putting this admin in the hands of a DAO helps reduce the centralization risk.

27. Misleading comments

Severity: Informational

In several instances of the codebase, there are inconsistencies where comments appear to be re-used from other functions and contain misleading information in the functions where they are tound:

- The `whitelist` function in `contracts/external/cw-tokenfactory-issuer/src/execute.rs:419-425` has incorrect comments that refer to blacklisting rather than whitelisting.
- The comment in `contracts/external/cw-tokenfactory-issuer/src/execute.rs:449` references the "change owner" functionality instead of the "force transfer" one.
- The `SetWhitelister` message documentation in `contracts/external/cw-tokenfactory-issuer/src/msg.rs:43` should refer to "whitelist addresses" instead of "blacklist addresses".
- The comment in `contracts/voting/dao-voting-cw721-staked/src/contract.rs:660` states that the minter is being updated, but the owner is updated instead.
- The error description in `contracts/voting/dao-voting-cw721-staked/src/error.rs:15` differs from the implementation, which allows thresholds less than or equal to 1.

- The error description in `contracts/voting/dao-voting-token-factory-staked/src/error.rs:22` differs from the implementation, which allows thresholds less than or equal to 1.
- The comment in `contracts/voting/dao-voting-token-factory-staked/src/msg.rs:37` states that `initial_balances` can not be empty. However, the implementation checks that the sum of the initial balance and `initial_dao_balance` is not zero.
- The error description in `contracts/voting/dao-voting-native-staked/src/error.rs:31` differs from the implementation, which allows thresholds less than or equal to 1.

Recommendation

We recommend reviewing the outlined comments to reflect the actual implementation.

Status: Resolved

28. "Migrate only if newer" pattern is not followed

Severity: Informational

The contracts within the scope of this audit are currently migrated without regard to their version. This can be improved by adding validation to ensure that the migration is only performed if the supplied version is newer.

Recommendation

We recommend following the "migrate only if newer" pattern defined in the CosmWasm documentation.

Status: Resolved

29. Contracts should implement a two step ownership transfer

Severity: Informational

The contracts within the scope of this audit allow the current owner to execute a one-step ownership transfer. While this is common practice, it presents a risk for the ownership of the contract to become lost if the owner transfers ownership to the incorrect address. A two-step ownership transfer will allow the current owner to propose a new owner, and then the account that is proposed as the new owner may call a function that will allow them to claim ownership and actually execute the config update.

Recommendation

We recommend implementing a two-step ownership transfer. The flow can be as follows:

1. The current owner proposes a new owner address that is validated.
2. The new owner account claims ownership, which applies the configuration changes.

Status: Resolved

30. Unused function in codebase

Severity: Informational

The `check_contract_has_funds` function in `contracts/external/cw-tokenfactory-issuer/src/helpers.rs:10` is not referenced anywhere else in the code base.

Although not a security issue, dead code decreases readability and maintainability.

Recommendation

We recommend removing unused code.

Status: Resolved

31. Freezing mechanism includes a bypass exception

Severity: Informational

The `cw-tokenfactory-issuer` contract contains a freezing mechanism controlled by the `IS_FROZEN` flag, which has been included for regulatory purposes, according to the client. However, this mechanism can be bypassed by any user added to `WHITELISTED_ADDRESSES`, creating an exception that might violate regulatory requirements.

Recommendation

We recommend removing the whitelist exception.

Status: Acknowledged

The client stated that this is intentional. For example, they might want to have a token that is non-transferable until the DAO is sufficiently decentralized. This functionality is intended for DAOs who do not wish to have their tokens liquid while bootstrapping their DAO. Another example is that a DAO may wish to allowlist a Token Staking contract (to allow users to stake their tokens in the DAO) or a Merkle Drop contract (to allow users to claim their tokens).