Defense By Thesis

Security Audit Report

Mezo

Mezo Earn Smart Contracts

Initial Report // January 8, 2026
Final Report // January 30, 2026

Team Members

Ahmad Jawid Jamiulahmadi // Senior Security Auditor
Mukesh Jaiswal // Senior Security Auditor

Table of Contents

About Thesis Defense

Defense is the security auditing arm of Thesis, Inc., the venture studio behind tBTC, Fold, Mezo, Acre, Taho, Etcher, and Embody. At Defense, we fight for the integrity and empowerment of the individual by strengthening the security of emerging technologies to promote a decentralized future and user freedom. Defense is the leading Bitcoin applied cryptography and security auditing firm. Our team of security auditors have carried out hundreds of security audits for decentralized systems across a number of ecosystems including Bitcoin, Ethereum + EVMs, Stacks, Cosmos SDK, NEAR and more. We offer our services within a variety of technologies including smart contracts, bridges, cryptography, node implementations, wallets and browser extensions, and dApps.

Defense will employ the Defense Audit Approach and Audit Process to the in scope service. In the event that certain processes and methodologies are not applicable to the in scope services, we will indicate as such in individual audit or design review SOWs. In addition, Thesis Defense provides clear guidance on successful Security Audit Preparation.

Section 1.0
Scope

Technical Scope

- **Repository:** https://github.com/mezo-org/tigris-token-launch
- **Audit Commit:** afafb4288e676363a8fea591c1d8020ec9d1d89e
- **Verification Commit:** 48e5705ffd51d2ac3a5a5c952f97558727f57a22

Section 2.0
Executive Summary

Schedule

This security audit was conducted from November 28, 2025 to December 31, 2025 by 2 senior security auditors for a total of 9 person-weeks.

System Overview

Mezo Earn is a dual-token voting mechanism combining locked Bitcoin (veBTC) with locked MEZO (veMEZO) to produce virtual voting weight. The design builds on Aerodrome's battle-tested evolution of the vote-escrow model, itself derived from Andre Cronje's Solidly ve(3,3) concept. The following outlines key innovations introduced beyond the established Aerodrome framework:

1. Dual-Token Governance: veBTC + veMEZO veBTC (locked BTC as NFTs) provides base voting weight; veMEZO (locked MEZO) acts as a multiplier. Governance influence requires holding both assets in proportion—preventing capture by either token class alone.
2. Gauge Architecture Users deploy Non-Staking Gauges that accept veBTC votes. veMEZO votes don't create independent weight—they multiply associated veBTC positions. This rewards long-term alignment while maintaining accessibility.
3. Emission Distribution (Splitter Gauges) Hierarchical allocation of post-rebase emissions: Block Splitter: Validators 20%, Chain Splitter 80% Chain Splitter: Staking Gauges 90% (LPs, MUSD Savings Rate), Non-Staking Gauges 10% (ecosystem/grants) Splitter allocations can shift max 1% per epoch, limiting governance attacks.
4. Boosting Logic Effective voting power = veBTC × min(5, 1 + Boost), where: Boost = 4 × (veBTC_total / veBTC_user) × (veMEZO_user / veMEZO_total) Multiplier ranges 1×–5×. Large veBTC holders need veMEZO to maximize influence.
5. Unvested Token Participation Grant recipients can lock unvested MEZO into veMEZO NFTs (cannot split/merge until vesting concludes). On revocation, unvested portion returns to grantor and NFT voting weight is immediately recalculated.

Security Audit Overview

We conducted a manual code review of the Mezo Earn implementation. The audit focused primarily on how the protocol's extensions and parameterizations of the Aerodrome design affect voting power accounting, boost mechanics, reward distribution, and their downstream effects. While the protocol retains core Aerodrome constructs such as locks, managed NFTs, and gauge-based voting, it introduces revocable grant-based veMEZO locks and virtual (boosted) voting power, which introduce discrete, non-monotonic state transitions and cached voting state that were not present in Aerodrome's original model and place additional stress on its lazy (poke-based) state propagation assumptions. In parallel, the protocol expands the system surface through additional voter types and expanded gauge categories, increasing integration complexity and the number of state interactions that rely on correct propagation. We reviewed how these changes interact with Voting Escrow state transitions, boost materialization, and reward accounting across epochs.

A central area of investigation was whether discrete state changes—such as grant revocation, managed NFT deposits and withdrawals, veNFT merges, and principal increases—are correctly reconciled with cached voting and boost state. We analyzed cases where voting power or boost eligibility is evaluated under one economic state and later reused after backing has changed, including scenarios involving revocable grants, managed veMEZO backing, and reuse of boost multipliers across veBTC principal-changing operations. Particular attention was paid to whether veMEZO backing is enforced as a single-consumer resource in real time and whether reductions in backing are synchronously reflected in all dependent boosted positions.

Finally, although protocol governance was explicitly out of scope, the audit considered how governance outcomes may be affected by voting power accounting and reward finalization mechanisms. This included reviewing the interaction between epoch-based reward distribution, non-retroactive reward finality, lazy reconciliation via pokes, and immutable governance snapshots. The objective was to determine whether temporarily stale or unsupported voting power could influence emissions or governance results before correction, and whether such behavior remains consistent with the protocol's intended economic and security assumptions.

Section 3.0
Key Findings Table

Issues	Severity	Status
ISSUE #1 Grant Revocation Leaves Stale Boost State Across veBTC NFTs	High	Acknowledged
ISSUE #2 Revocable Grants Are Credited with Long-Duration Voting Power Without Realized Commitment	High	Acknowledged
ISSUE #3 Boost Is Not Revalidated on veBTC Principal-Changing Operations	High	Fixed
ISSUE #4 Single veMEZO NFT Can Back Multiple veBTC Boosts Concurrently	High	Acknowledged
ISSUE #5 Boosted veBTC Voting Power Is Not Updated After Managed veMEZO Withdrawal	High	Fixed
ISSUE #6 veNFTs With Active Gauge Votes Can Be Deposited Into Managed veNFTs	High	Fixed
ISSUE #7 Precision Loss Leads to Trapped/Lost Yield During Distribution	Medium	Fixed
ISSUE #8 `__VotingEscrow_initialize` Function Lacks the `onlyInitializing` Protection	Low	Fixed
ISSUE #9 Zero Address Grant Manager Prevents Grant Revocation	Low	Fixed

Severity definitions can be found in Appendix A

Section 4.0
Findings

We describe the security issues identified during the security audit, along with their potential impact. We also note areas for improvement and optimizations in accordance with best practices. This includes recommendations to mitigate or remediate the issues we identify, in addition to their status before and after the fix verification.

ISSUE#1

Grant Revocation Leaves Stale Boost State Across veBTC NFTs

High   Acknowledged

Description

The protocol allows grant managers to create revocable veMEZO grant locks for grantees. These veMEZO locks can be used to boost the voting power of veBTC NFTs up to a fixed multiplier (5x). When a grant is revoked, the veMEZO lock amount is immediately reduced (potentially to near-zero). However, any veBTC NFTs whose boost depends on the revoked veMEZO grant are not updated at revocation time. Instead, the protocol relies on a later "poke" to propagate the change. Until such propagation occurs, veBTC NFTs continue to participate in gauge voting, reward distribution, and protocol governance using stale boosted voting power that no longer has economic backing. This creates a staleness window after grant revocation in which rewards and governance influence are computed using invalid state.

For gauge voting and reward distribution, this allows unbacked voting power to influence emissions and rewards until the affected veBTC NFT is poked. If a revocation occurs shortly before the voting cutoff at the end of an epoch, the remaining window for a corrective poke is limited, increasing the likelihood that the epoch is finalized using stale voting power. For protocol-wide governance, voting relies on a single, immutable snapshot of voting power per proposal. If a grant is revoked before the governance snapshot but the dependent veBTC boost is not yet propagated, the snapshot permanently records boosted voting power that no longer has economic backing. Unlike gauge voting, this error cannot be corrected by later pokes and directly affects proposal outcomes.

The system models grant revocation as equivalent to time-based voting power decay and relies on poke-based eventual consistency to propagate the change. This assumption is incorrect for discrete state transitions that immediately remove economic backing. Time-based decay is continuous, affects all participants symmetrically, and justifies lazy propagation. Grant revocation is a discrete, transactional event with asymmetric effects; if not synchronously propagated, it allows boosted positions to retain voting power they are no longer economically entitled to. Applying the same poke-based model to both continuous decay and discrete grant revocation therefore introduces a correctness gap in reward distribution and governance accounting.

Note: While this issue focuses on the effect of grant revocation on veBTC boost backing, revocable veMEZO locks may also be used directly for gauge voting. As a result, the same staleness window can allow revoked grants to influence other gauges beyond boost-related flows until state is corrected.

Impact

This issue results in incorrect reward distribution and governance influence based on voting power that no longer has economic backing. Although users cannot create grants for themselves, the issue remains exploitable under realistic threat models, including grant manager compromise, collusion, or administrative error.

Scenario 1 — Grant Manager Abuse or Collusion

A malicious or compromised grant manager can:

1. Create a large veMEZO grant for a controlled or colluding grantee.
2. Use the grant to boost one or more veBTC NFTs to the maximum multiplier.

3. Use the boosted veBTC NFTs to vote into high-emission gauges and to participate in protocol governance.
4. Revoke the grant immediately, reclaiming the underlying MEZO.
5. Allow the boosted veBTC voting power to remain active until a poke occurs.

For gauge voting, rewards are distributed using unbacked boosted voting power, diluting honest participants. Rewards accrued during this staleness window are irreversible.

For protocol governance, the impact is more severe. By timing grant creation and revocation around governance snapshots, the grant manager can cause temporarily boosted voting power to be snapshotted, allowing proposals to pass or fail based on voting power that no longer has economic backing. Once the snapshot is taken, subsequent revocations or pokes cannot retroactively correct the outcome.

By timing revocations shortly before the voting cutoff at the end of an epoch, a malicious actor can limit the window for corrective pokes, increasing the likelihood that an epoch is finalized using stale boosted voting power and making the resulting reward and governance impact attacker-controlled.

This enables governance influence to be obtained with minimal effective capital lock-up and allows the same MEZO capital to be reused across multiple proposals or epochs before correction occurs.

Scenario 2 — Operational or Administrative Revocation

Even in the absence of malicious intent:

- A grant may be revoked for operational, compliance, or administrative reasons.
- The protocol immediately enters a state where boosted veBTC voting power is economically invalid but remains active.
- Rewards, emissions, and governance influence continue to be computed incorrectly until an external poke is triggered.

In all scenarios, correctness depends on external actors detecting revocations and proactively triggering pokes. This model provides only eventual consistency and is insufficient for reward distribution and governance mechanisms that assume immediately correct economic state.

Recommendation

Instead of immediately reducing the revoked grant’s lock amount by the unvested portion and transferring the underlying MEZO to the grant manager, we recommend preserving the economic backing by atomically splitting the grant lock at revocation: reduce the grantee’s veMEZO lock amount by the revoked portion and mint an equivalent veMEZO lock for the grant manager with the same unlock time.

The newly created lock must be ineligible for use as boost backing or for gauge voting until all veBTC boosts and gauge votes that depended on the revoked grant have been settled.

Verification Status

The Mezo team stated that grants can be created and revoked only by grant managers, who are trusted system participants appointed by governance. As with governance itself, grant managers are expected to use appropriate security practices, such as multisignature wallets with sufficiently high thresholds. Grant revocations are expected to be rare and reserved for exceptional circumstances.

ISSUE#2

Revocable Grants Are Credited with Long-Duration Voting Power Without Realized Commitment

High

Acknowledged

Description

The protocol supports revocable veMEZO grant locks created by a grant manager for a grantee. These grants can be configured with long lock durations (e.g., four years) and are treated identically to non-revocable locks for voting power and boosting, including their ability to boost veBTC voting power.

Revocable grants receive full long-duration voting power immediately, based solely on the configured lock duration, and may also be configured as permanent locks immediately after creation for maximum voting power. If a grant is revoked early, voting power is reduced only prospectively. Any voting power, rewards, or governance influence exercised prior to revocation is not adjusted or clawed back, even though the revoked amount was not ultimately committed for the full lock duration.

Voting power therefore reflects the maximum possible commitment, not the realized commitment of revocable capital. The protocol does not differentiate between revocable and non-revocable locks when assigning voting power and provides no mechanism to discount, vest, or otherwise condition voting power on continued lock commitment.

This behavior is independent of propagation delays or poke-based mechanisms. The inconsistency arises even with immediate state updates and reflects an economic design mismatch: revocable capital is granted long-term voting power upfront despite the possibility of early revocation.

Impact

This design results in irreversible misallocation of rewards and governance influence. Revocable grants receive full long-duration or perpetual voting power immediately and compete as if permanently committed, even though part or all of the grant may later be revoked. When revocation occurs, only future voting power is reduced; rewards and governance influence exercised prior to revocation are not corrected.

Under normal operation, this causes systematic dilution of honest long-term lockers and permanently skews emissions and governance outcomes based on capital that did not ultimately remain locked.

A malicious or compromised grant manager can further amplify this effect by intentionally issuing large long-duration or perpetual revocable grants, allowing them to accrue rewards and governance influence for a limited period, and then revoking them once sufficient influence has been exercised. This enables repeated extraction of long-term voting power benefits without sustaining the corresponding long-term lock commitment.

As a result:

- Emissions and governance outcomes reflect voting power inconsistent with realized economic commitment.
- Honest participants are diluted by capital that bears lower opportunity cost.
- The distortion is irreversible, as past rewards and governance decisions cannot be clawed back.

This issue arises from economic design and persists regardless of propagation timing, poking behavior, or synchronization guarantees.

Recommendation

Instead of immediately reducing the revoked grant's lock amount by the unvested portion and transferring the underlying MEZO to the grant manager, we recommend atomically splitting the grant lock at revocation: reduce the grantee's veMEZO lock amount by the revoked portion and mint an equivalent veMEZO lock for the grant manager with the same unlock time. The newly created lock must be ineligible for use as boost backing or for gauge voting until all veBTC boosts and gauge votes that depended on the revoked grant have been settled.

Verification Status

The Mezo team stated that grants can be created and revoked only by grant managers, who are trusted system participants appointed by governance. As with governance itself, grant managers are expected to use appropriate security practices, such as multisignature wallets with sufficiently high thresholds. Grant revocations are expected to be rare and reserved for exceptional circumstances.

ISSUE#3

Boost Is Not Revalidated on veBTC Principal-Changing Operations

High ☑ Fixed

Location

contracts/ve/Escrow.sol#L562-L581

contracts/ve/Escrow.sol#L72-L90

contracts/ve/Escrow.sol#L712-L737

contracts/ve/ManagedNFT.sol#L112-L132

Description

Boosted veBTC NFTs rely on a cached boost multiplier that is applied to voting power and reward accounting. This multiplier is intended to reflect the relationship between veBTC voting power and veMEZO backing at a given point in time. However, in the `VotingEscrow` smart contract, the protocol reuses this cached boost across multiple state transitions that materially change the boosted position's economic characteristics.

Specifically:

- In the `_increaseAmountFor` function, the newly added veBTC principal is multiplied by the existing boost without recomputing boost eligibility.
- In the `merge` function, the veBTC principal from one NFT is merged into another NFT that already carries a boost, causing the merged balance to inherit the existing multiplier.
- In the `depositManaged` function, managed veMEZO voting power changes dynamically through deposits, yet any veBTC NFT boosted by the managed position is not synchronously updated to reflect the new backing.

Across these flows, principal or backing changes occur without immediate re-computation or capping of boost. As a result, boost eligibility is evaluated under one state and then reused under a materially different state, allowing newly added or merged balances to benefit from boost levels that are not economically supported.

Continuous time-based decay is intentionally handled via lazy propagation. By contrast, discrete veBTC principal-changing transitions require immediate revalidation, as reusing cached boost multipliers across such transitions can result in unsupported boosted voting power.

Note: veMEZO NFTs include a boost parameter that currently remains fixed at 1. Should veMEZO-level boosting be introduced in the future, the same stale attribution risks would apply to boosted veMEZO voting power.

Impact

This issue enables systematic amplification of boosted voting power without proportional economic backing.

An attacker can establish a high boost under favorable conditions (e.g., small veBTC amount or temporary backing) and then expand the boosted position via increases, merges, or managed deposits, causing large balances to inherit the boost without providing the required veMEZO support.

If the inflated boost persists through epoch finalization, rewards and emissions are irreversibly misallocated. If a governance snapshot occurs during the staleness window, proposal outcomes may be determined using overstated voting power that cannot be corrected retroactively.

Because these flows rely on cached boost state rather than invariant enforcement, they undermine the core economic guarantees of the boosting system and enable repeated extraction of boost benefits at significantly reduced cost, affecting both reward distribution and governance outcomes.

Recommendation

We recommend re-computing or capping the boost applied to a veBTC NFT on any discrete veBTC principal-changing transition, including increasing, merging, and depositing into a managed veBTC position. Newly added or restructured veBTC principal must not inherit a cached boost multiplier without revalidation.

Verification Status

The fix implemented by the Mezo team correctly ensures that boost state is synchronously recomputed on veBTC principal-changing operations, eliminating stale boost attribution in those flows. However, gauge vote state continues to be not updated synchronously and relies on external actors via pokes for eventual consistency. As a result, while boost correctness is improved at the veBTC level, gauge voting and reward attribution may still temporarily reflect outdated state until reconciliation occurs.

ISSUE#4

Single veMEZO NFT Can Back Multiple veBTC Boosts Concurrently

High

Acknowledged

Location

contracts/BoostVoter.sol#L166-L170

contracts/VotingEscrow.sol#L426-L428

Description

The protocol allows a single veMEZO NFT to be used as boost backing for veBTC NFTs via gauge voting. A veBTC NFT becomes boosted when a veMEZO NFT votes on its associated boost gauge and the boost is materialized through a `pokeBoost`, which triggers `updateBoost` for the veBTC NFT.

Once a veBTC NFT's boosted voting power is materialized, it persists until the veBTC NFT is explicitly poked again, regardless of subsequent changes to the veMEZO NFT's gauge votes. In particular, a veMEZO NFT can:

1. Vote on a boost gauge associated with veBTC NFT A.
2. Trigger `pokeBoost` so that veBTC A receives boosted voting power.
3. Reset or change its gauge vote in a later epoch.
4. Repeat the process for veBTC NFT B.

The previously boosted veBTC NFT A retains its boosted voting power until it is explicitly updated, even though the veMEZO NFT is no longer voting for it. As a result, the same veMEZO NFT can transiently back multiple veBTC boosts concurrently.

This behavior is exacerbated by the protocol's reliance on poke-based, off-chain–driven updates. By timing the reset and re-vote shortly before the voting cutoff in the final hour of an epoch, a malicious user can limit the window for a corrective poke to as little as just over one hour, increasing the likelihood that the previously boosted veBTC NFT is not updated before epoch finalization. This allows the staleness window to extend across epoch boundaries, during which multiple veBTC NFTs can appear simultaneously boosted by the same veMEZO backing.

Beyond reward accounting, this staleness also affects protocol governance. Governance voting relies on immutable snapshots of voting power. If a governance snapshot is taken while multiple veBTC NFTs reflect boosted voting power derived from the same veMEZO backing, the snapshot permanently records voting power that exceeds the economically backed state. Subsequent pokes cannot retroactively correct governance outcomes.

As a result, boost backing is not enforced as a single-consumer resource in real time, and boosted voting power can temporarily exceed the amount supported by underlying veMEZO voting power.

Impact

This issue allows a single veMEZO NFT to be used to back boosted voting power for multiple veBTC NFTs concurrently, resulting in duplicated boost effects across rewards and governance.

A malicious user can deliberately sequence gauge votes, resets, and `pokeBoost` calls so that multiple veBTC NFTs retain boosted voting power simultaneously, despite only one veMEZO NFT providing backing.

During this staleness window:

- Multiple veBTC NFTs benefit from boost multipliers that are not economically justified.
- Gauge emissions and rewards are distributed based on inflated boosted voting power, diluting honest participants.
- Governance voting power can be overstated if proposal snapshots occur during the staleness window.

Because boost effects persist until explicitly recomputed and poke execution depends on external actors or bots, this behavior can be reliably engineered, particularly near epoch boundaries where the window for corrective updates is bounded. Rewards distributed during this period are irreversible, and governance outcomes influenced by the excess boosted voting power cannot be corrected retroactively.

Overall, the protocol permits transient but exploitable double-use of boost backing, violating the assumption that each veMEZO NFT can back at most one boosted position at a time and undermining the economic and governance integrity of the boost mechanism.

Recommendation

To ensure that a veMEZO NFT cannot simultaneously back boosted voting power for more than one veBTC NFT, we recommend enforcing single-consumer semantics for boost backing. Specifically:

- When a veMEZO NFT changes or resets its boost gauge vote, all veBTC NFTs whose boosted state depends on that veMEZO NFT should be synchronously updated or invalidated before the veMEZO NFT can be reused as boost backing elsewhere; or
- The protocol should explicitly reserve a veMEZO NFT as boost backing for a single veBTC NFT at a time, preventing it from being used to materialize new boosts or to vote on any other gauges until the previously boosted veBTC NFT has been updated.

Verification Status:

The Mezo team stated that the use of poke-based, off-chain–driven updates instead of synchronous updates is an intentional design choice. This approach avoids imposing strict limitations on the number of veBTC gauges that veMEZO holders can vote on, while preserving the same user experience as pool voting.

During the final hour, both gauge vote pokes and boost pokes remain possible. Voting itself is disabled during this period unless the veNFT has been explicitly allowlisted by governance. This provides the poke maintainer with at least one hour to execute corrective boost pokes.

Moreover, executing a full chain of upgrades is expensive and cannot be fully enforced on-chain. Each veBTC is eligible to vote with three voters - the Pools Voter, the Ecosystem Voter, and the Validator Voter - each supporting voting on 30 gauges by default. Even if boost and veBTC voting power recalculations were enforced on-chain, resetting all previously cast votes would still require separate poke transactions to avoid the risk of exceeding block size limits.

To address these constraints, the on-chain system is complemented by an off-chain maintainer that pokes boosts and votes based on a defined set of heuristics, prioritizing actions with the highest impact. The voting system is deployed on the Mezo chain, and we plan to introduce a dedicated, chain-integrated mechanism that will allow maintainers to execute these pokes automatically at zero cost.

ISSUE#5

Boosted veBTC Voting Power Is Not Updated After Managed veMEZO Withdrawal

High Fixed

Location

contracts/NonStakingVoter.sol#L353

Description

In the BoostVoter flow, the `withdrawManaged` function in the `NonStakingVoter` smart contract allows a veMEZO NFT (`_tokenId`) to be withdrawn from a managed veMEZO position (`_mTokenId`). This withdrawal reduces the managed veMEZO's effective locked amount and voting power. However, it does not ensure that any veBTC NFT currently boosted by `_mTokenId` is synchronously `pokeBoost` ed. As a result, the veBTC NFT may continue to operate with a boost computed from the pre-withdrawal (higher) veMEZO voting power.

This creates a staleness window in which boosted voting power no longer reflects current backing. During this window, the withdrawn veMEZO NFT can be reused elsewhere, while the veBTC NFT continues benefiting from an overstated boost.

Although managed operations are restricted during the final hour of an epoch, this only bounds the timing window and does not eliminate the staleness period. A withdrawal executed shortly before the vote cutoff can allow the epoch to be finalized using an overstated boost value, after which rewards are distributed irreversibly, regardless of whether the boost is corrected later.

Governance voting is not epoch-scoped; if a governance proposal snapshot occurs during the staleness window, the snapshot may capture overstated boosted voting power, and subsequent `pokeBoost` calls cannot retroactively correct the recorded voting weight or proposal outcome.

Impact

This issue allows veBTC NFTs to temporarily retain inflated boosted voting power after the effective veMEZO backing has been reduced, resulting in incorrect reward distribution and governance influence. During the staleness window following a `withdrawManaged` operation:

- A veBTC NFT may continue to benefit from a boost computed using pre-withdrawal veMEZO voting power, even though that backing no longer exists.
- The withdrawn veMEZO NFT can be reused elsewhere, enabling the same economic weight to influence multiple positions while the original boosted veBTC remains overstated.
- Gauge emissions and rewards may be allocated based on boosted voting power that exceeds the current economic backing.

If the withdrawal is executed shortly before the vote cutoff, the affected epoch can be finalized using the overstated boost value. Rewards distributed for that epoch are irreversible and cannot be corrected even if the boost is recomputed later.

Separately, governance voting is not epoch-scoped. If a governance proposal snapshot occurs during the staleness window, the snapshot may record overstated boosted voting power, permanently affecting proposal outcomes. Subsequent `pokeBoost` calls cannot retroactively adjust the recorded voting weight.

Recommendation

We recommend ensuring that boost state is synchronously updated whenever the backing of a managed veMEZO NFT changes. Specifically, the `withdrawManaged` function must trigger `pokeBoost` for the affected managed veMEZO NFT (`_mTokenId`), in addition to any generic poke, to ensure that boosted voting power is immediately recalculated based on the updated backing.

This prevents managed veMEZO NFTs from retaining stale or inflated boost values after withdrawals and ensures that withdrawn veNFTs cannot be reused to boost other veBTC NFTs while the original managed position still benefits from outdated boost state.

Verification Status

The fix implemented by the Mezo team correctly ensures that boost state is synchronously recomputed on withdrawal from managed NFTs, eliminating stale boost attribution in this flow. However, gauge vote state continues to be not updated synchronously and relies on external actors via pokes for eventual consistency. As a result, while boost correctness is improved at the veBTC level, gauge voting and reward attribution may still temporarily reflect outdated state until reconciliation occurs.

ISSUE#6

veNFTs With Active Gauge Votes Can Be Deposited Into Managed veNFTs

^ High ☑ Fixed

Location

contracts/Voter.sol#L350

contracts/NonStakingVoter.sol#L331

Description

The `depositManaged` functions in the `Voter` and `NonStakingVoter` smart contracts allow a veNFT (`_tokenId`) to be deposited into a managed veNFT (`_mTokenId`). However, these functions do not verify that the veNFT being deposited has no active gauge votes at the time of deposit.

As a result, a veNFT that has already voted in one or more gauges can be deposited into a managed NFT without first clearing or updating its existing gauge votes. The voting effects of those votes remain active until the veNFT is explicitly poked, even though ownership and voting control over the veNFT have effectively changed as a result of the deposit.

Because gauge vote state is updated lazily via poke, this creates a staleness window in which the deposited veNFT continues to influence gauge voting and reward distribution based on outdated state. If the `depositManaged` function is executed shortly before the voting cutoff in the final hour of an epoch, the remaining window for a corrective poke is constrained, increasing the likelihood that the epoch is finalized with a stale voting state still active.

As a result, gauge voting effects can persist beyond their intended scope, despite the veNFT no longer being independently controlled.

Impact

This issue allows veNFTs to continue influencing gauge voting and reward distribution after being deposited into managed NFTs, based on gauge vote state that should no longer be effective. During the staleness window:

- Gauge emissions and rewards may be allocated using voting power that is no longer correctly attributable.
- Honest participants are diluted by voting influence that persists beyond the intended ownership and control boundaries.
- The effect can be reliably amplified through timing, particularly by executing deposits near epoch boundaries.

Because gauge rewards distributed at epoch finalization are irreversible, any emissions allocated using stale voting state cannot be corrected retroactively. Correctness therefore depends on external actors detecting the stale state and proactively calling poke, which provides only eventual consistency and is insufficient for reward distribution mechanisms that assume immediate correctness upon changes in voting ownership or control.

Recommendation

Before allowing a veNFT (`_tokenId`) to be deposited into a managed NFT (`_mTokenId`), we recommend verifying that the veNFT has no active gauge votes. If the veNFT has participated in any gauge voting, the deposit should be rejected.

This ensures that a veNFT cannot carry stale or externally attributable voting power into a managed position and avoids reliance on poke-based cleanup after the deposit.

ISSUE#7

Precision Loss Leads to Trapped/Lost Yield During Distribution

= Medium ☑ Fixed

Location

contracts/vaults/MUSDSavingsRate.sol#L245

Description

The `_receiveYield` function updates the `yieldIndex` using: `ratio = (amount * 1e18) / totalSupply` . When the vault's `totalSupply` is very large, small yield amounts can be rounded down to zero due to truncation.

Impact

Any yield distribution below a certain ratio threshold is effectively lost due to integer division. For example, when the total supply is 1 billion MUSD, any strategy yield contribution of 0.99 gwei or less is rounded down to zero and permanently discarded from the system's accounting.

Recommendation

We recommend tracking leftover yield in a separate variable and adding it to the next yield update to prevent loss due to rounding.

ISSUE#8

`__VotingEscrow_initialize` Function Lacks the `onlyInitializing` Protection

✓ Low ☑ Fixed

Location

contracts/VotingEscrow.sol#L59

Description

The `VotingEscrow` smart contract defines an internal initialization function, `__VotingEscrow_initialize`, which sets critical protocol parameters, including the underlying token address and `maxLockTime`.

While this function is currently marked as `internal` and is invoked by the child contract's `initialize` function using the `initializer` modifier, it does not use the `onlyInitializing` modifier. According to OpenZeppelin's upgradeable contract standards, all internal functions that participate in the initialization chain should be protected by `onlyInitializing` to ensure they can only be executed during the initialization phase.

Impact

The absence of the `onlyInitializing` modifier introduces a risk of unintended state mutation in complex inheritance scenarios or future upgrades. If a future contract version or another child contract inadvertently exposes a callable path to `__VotingEscrow_initialize` after the proxy has already been initialized, critical protocol state could be overwritten or corrupted.

Recommendation

We recommend adding the `onlyInitializing` modifier to the `__VotingEscrow_initialize` function in `VotingEscrow` smart contract.

ISSUE#9

Zero Address Grant Manager Prevents Grant Revocation

✓ Low ☑ Fixed

Location

contracts/ve/Grant.sol#L89-L100

contracts/ve/Grant.sol#L107-L109

Description

The `_setGrantManager` function in the `Grant` smart contract does not validate the `_newGrantManager` parameter against zero address. As a result, the grant manager for a given `_tokenId can be set to address(0)`.

Once `self.grantManager[_tokenId]` is set to the zero address, the grant becomes effectively non-revocable. Both `_setGrantManager` and `_revokeGrant` functions require an active (non-zero) grant manager to be callable, and no alternative mechanism exists to recover or revoke the grant after the grant manager is cleared.

Impact

If the grant manager is set to the zero address, the associated grant becomes permanently irrevocable. As a result, locked MEZO associated with the grant may incorrectly be assumed to be revocable, resulting in loss of control over grant lifecycle management.

Recommendation

We recommend validating the `_newGrantManager` parameter in the `_setGrantManager` function to ensure it is not zero address.

Verification Status

The Mezo team stated that this is an intentional design choice that enables the creation of non-revocable grants. The project documentation has been updated to clearly describe this mechanism.

Section 5.0
Appendix A

At Thesis Defense, we utilize the Immunefi Vulnerability Severity Classification System - v2.3.

Severity	Definition
Critical	• Manipulation of governance voting result deviating from voted outcome and resulting in a direct change from intended effect of original results • Direct theft of any user funds, whether at-rest or in-motion, other than unclaimed yield • Direct theft of any user NFTs, whether at-rest or in-motion, other than unclaimed royalties • Permanent freezing of funds • Permanent freezing of NFTs • Unauthorized minting of NFTs • Predictable or manipulable RNG that results in abuse of the principal or NFT • Unintended alteration of what the NFT represents (e.g. token URI, payload, artistic content) • Protocol insolvency
High	• Theft of unclaimed yield • Theft of unclaimed royalties • Permanent freezing of unclaimed yield • Permanent freezing of unclaimed royalties • Temporary freezing of funds • Temporary freezing NFTs
Medium	• Smart contract unable to operate due to lack of token funds • Enabling/disabling notifications • Griefing (e.g. no profit motive for an attacker, but damage to the users or the protocol) • Theft of gas • Unbounded gas consumption
Low	• Contract fails to deliver promised returns, but doesn't lose value
None	• We make note of issues of no severity that reflect best practice recommendations or opportunities for optimization, including, but not limited to, gas optimization, the divergence from standard coding practices, code readability issues, the incorrect use of dependencies, insufficient test coverage, or the absence of documentation or code comments.

Section 6.0
Appendix B

Thesis Defense Disclaimer

Thesis Defense conducts its security audits and other services provided based on agreed-upon and specific scopes of work (SOWs) with our Customers. The analysis provided in our reports is based solely on the information available and the state of the systems at the time of review. While Thesis Defense strives to provide thorough and accurate analysis, our reports do not constitute a guarantee of the project's security and should not be interpreted as assurances of error-free or risk-free project operations. It is imperative to acknowledge that all technological evaluations are inherently subject to risks and uncertainties due to the emergent nature of cryptographic technologies.

Our reports are not intended to be utilized as financial, investment, legal, tax, or regulatory advice, nor should they be perceived as an endorsement of any particular technology or project. No third party should rely on these reports for the purpose of making investment decisions or consider them as a guarantee of project security.

Links to external websites and references to third-party information within our reports are provided solely for the user's convenience. Thesis Defense does not control, endorse, or assume responsibility for the content or privacy practices of any linked external sites. Users should exercise caution and independently verify any information obtained from third-party sources.

The contents of our reports, including methodologies, data analysis, and conclusions, are the proprietary intellectual property of Thesis Defense and are provided exclusively for the specified use of our Customers. Unauthorized disclosure, reproduction, or distribution of this material is strictly prohibited unless explicitly authorized by Thesis Defense. Thesis Defense does not assume any obligation to update the information contained within our reports post-publication, nor do we owe a duty to any third party by virtue of making these analyses available.