Altitude Unit Audit Report

Prepared by T1MOH

Version 1.0

Lead Auditors

T1MOH

September 14th, 2026

Contents

1 About T1MOH

T1MOH is an independent smart contract security researcher working with Spearbit, Cyfrin, Sherlock, BurraSec.

Portfolio can be found by link.

2 Disclaimer

T1MOH makes every effort to find as many vulnerabilities in the code as possible in the given time but holds no responsibility for the findings in this document. A security audit does not endorse the underlying business or product. The audit was time-boxed and the review of the code was solely focused on the security aspects of the Solidity implementation of the contracts.

3 Risk Classification

	Impact: High	**Impact: Medium**	**Impact: Low**
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4 Protocol Summary

The reviewed contracts include:

- `Unit` - the `unitUSD` ERC20 token (6 decimals) with role-based access control. A `MINTER_ROLE` can mint, holders can burn, and `DEFAULT_ADMIN_ROLE` can freeze/unfreeze accounts and confiscate balances from frozen accounts.
- `StakedUnit` - a non-transferable ERC4626 vault that lets users stake `unitUSD` for `sUNIT` shares.
- `Minter2` - the entry point for users. It mints `unitUSD` against deposited `USDT` and processes redemptions, with redemptions gated by an off-chain AML check.
- `CumulativeMerkleDrop` - an `Ownable2Step` distribution contract that pays out tokens based on a cumulative Merkle root, allowing accounts to claim the difference between their cumulative allocation and what they have already claimed.

5 Audit Scope

5.0.1 unit-core

Contract
`src/Unit.sol`
`src/StakedUnit.sol`
`src/Minter2.sol`
`src/CumulativeMerkleDrop.sol`

6 Executive Summary

Over the course of 1 day, T1MOH conducted an audit on the Unit smart contracts provided by Altitude. In this period, a total of 1 issues were found.

The findings consist of 1 Informational issue.

Summary

Repository	unit-core
Commit	09627be72029...
Audit Timeline	Sep 14th, 2026
Methods	Manual Review

Issues Found

Critical Risk	0
High Risk	0
Medium Risk	0
Low Risk	0
Informational	1
Gas Optimizations	0
Total Issues	1

Summary of Findings

Admin bypasses freeze checks in `Unit._update`, allowing transfers to a frozen address	Acknowledged

7 Findings

7.1 Informational

7.1.1 Admin bypasses freeze checks in `Unit._update`, allowing transfers to a frozen address

Description: In `Unit._update`, the freeze checks are entirely skipped when the call is initiated by an account holding `DEFAULT_ADMIN_ROLE`:

```
function _update(address from, address to, uint256 value) internal override {
    if (!hasRole(DEFAULT_ADMIN_ROLE, msg.sender)) {
        if (from != address(0) && isFrozen[from]) revert AccountFrozen();
        if (to != address(0) && isFrozen[to]) revert AccountFrozen();
    }
    super._update(from, to, value);
}
```

This bypass is required by design: `confiscate()` calls `_transfer(from, to, value)`, where `from` is typically the frozen account itself. However, the bypass is implemented more broadly than `confiscate()` requires: it removes the check not only for `from` but also for `to`, and it applies to any transfer initiated by the admin, not only the confiscation path.

Recommended Mitigation: Consider acknowledging. There is no impact, so no need to complicate existing logic. Slight acknowledging comment is enough.

UNIT: Acknowledged.