
# Metaone Audit Report

January 2026

# Metaone Audit Report

Zero Drift

January 2026

Prepared by: Zero Drift

## Table of Contents



## About Zero Drift

Zero Drift is a tech-driven Web3 auditing firm dedicated to zero-incident smart-contract assurance for mission-critical protocols.

Our team consists of several senior smart-contract auditors, each with 5+ years of experience and a perfect post-audit record across more than 80 successful reviews, collectively helping protect over $5 billion in on-chain assets.

## Disclaimer

The Zero Drift team makes all effort to find as many vulnerabilities in the code in the given time period, but holds no responsibilities for the findings provided in this document. A security audit by the team is not an endorsement of the underlying business or product. The audit was time-boxed and the review of the code was solely on the security aspects of the smart-contract implementation of the contracts.

## Risk Classification

| | Impact: High | Impact: Medium | Impact: Low |
|---|---|---|---|
| Likelihood: High | Critical | High | Medium |
| Likelihood: Medium | High | Medium | Low |
| Likelihood: Low | Medium | Low | Low |

## Protocol Overview

The Metaone codebase under review contains a DailyLogin mechanism that tracks per-user login counts by day relative to a configurable `startTime`, and emits events on successful logins. The design emphasizes simplicity and minimal state: users can log in once per day, their last-login day is recorded, and an event reflects the login activity.

## Audit Scope

The scope consisted of a focused review of the DailyLogin logic and auxiliary configuration details reported by the client.

- Files reviewed: `Metaone.sol` (commit `92203872f69e0b746a6160cccd9bfa80db2933b6` ), references to `DailyLogin.sol`

## Executive Summary

Over the course of the review, the Zero Drift team examined the DailyLogin functionality and associated patterns. We identified several issues across High, Medium, and Low severities. The project team has

addressed all identified issues; each finding below is marked Resolved.

**Summary**

| Project Name | Metaone |
|---|---|
| Repository | https://github.com/METAONEOFFICAL/Metaone/blob/main/login.sol |
| Commit | 92203872f69e0b746a6160cccd9bfa80db2933b6 |
| Audit Timeline | Jan 10–12, 2026 |
| Methods | Manual Review |

**Issues Found**

| | Count |
|---|---|
| Critical Risk | 0 |
| High Risk | 1 |
| Medium Risk | 3 |
| Low Risk | 5 |
| Informational | 0 |
| Gas Optimizations | 0 |
| **Total Issues** | **9** |

**Summary of Findings**

| ID | Title | Severity | Status |
|---|---|---|---|
| H-1 | Redundant access control implementation (Ownable unused) | High | Resolved |
| M-1 | Incorrect event emission order | Medium | Resolved |
| M-2 | Inefficient duplicate calculation of current day | Medium | Resolved |
| M-3 | Overly broad compiler pragma range | Medium | Resolved |

| ID | Title | Severity | Status |
|---|---|---|---|
| L-1 | Hardcoded time constant (magic number) | Low | Resolved |
| L-2 | Missing view function for current-day calculation | Low | Resolved |
| L-3 | Grammatical error in error message | Low | Resolved |
| L-4 | Inconsistent inequality check at start time | Low | Resolved |
| L-5 | Unused math library import | Low | Resolved |

## Findings

### High Risk

#### [H-01] Redundant Access Control Implementation

Description

The contract inherits from `Ownable` without using any owner-restricted functionality, adding unnecessary complexity and gas overhead. In addition, the constructor uses deprecated/invalid `Ownable` initialization syntax that will fail to compile with modern OpenZeppelin versions.

Recommendation

Remove the `Ownable` import and inheritance if not required; avoid unused admin patterns. If ownership is needed in the future, implement explicit access controls and use current OpenZeppelin patterns.

Status: Resolved

### Medium Risk

#### [M-01] Incorrect Event Emission Order

Impact: Event logs report stale login counts, diverging from on-chain state and potentially breaking downstream consumers.

Description

`Login` is emitted before incrementing the user's login count, causing the event to reflect the previous value.

Recommendation

Increment the state first, then emit the event using the updated value.

Status: Resolved

#### [M-02] Inefficient Duplicate Calculation

Impact: Unnecessary gas usage and potential for inconsistency if the expression is duplicated and later modified unevenly.

Description

The expression `(block.timestamp - startTime)/ (3600 * 24)+ 1` is computed multiple times within a single path.

Recommendation

Compute once (e.g., via `currentDay` or `getCurrentDay()`) and reuse.

Status: Resolved

#### [M-03] Unspecified Compiler Compatibility

Impact: An overly broad pragma range spanning multiple breaking changes risks compilation failures or subtle bugs depending on the chosen compiler.

Description

The pragma `>=0.4.22 <0.9.0` is excessively broad.

Recommendation

Use a narrow, tested range, e.g., `>=0.8.0 <0.9.0` with explicit compiler settings aligned to the toolchain.

Status: Resolved

### Low Risk

#### [L-01] Hardcoded Time Constants

Description

Using `3600 * 24` decreases readability and increases error risk.

Recommendation

Use Solidity time units (e.g., `1 days`) or a named constant.

Status: Resolved

### [L-02] Missing View Function for Day Calculation

Description

No public view function exposes the current day calculation, forcing off-chain duplication.

Recommendation

Add a `getCurrentDay()` view function to standardize the logic and improve integrability.

Status: Resolved

### [L-03] Grammatical Error in Error Message

Description

Error message reads “Daily login start soon” instead of a correct phrasing.

Recommendation

Change to “Daily login has not started yet”.

Status: Resolved

### [L-04] Inconsistent Inequality Check

Description

Using a strict > for `block.timestamp > startTime` excludes the exact start timestamp.

Recommendation

Use >= so the exact start moment is included.

Status: Resolved

### [L-05] Unused Math Library Import

Description

`using Math for uint` is declared but unused.

Recommendation

Remove unused imports and `using` declarations to reduce bytecode size and improve clarity.

Status: Resolved