# Offchain Labs - Sequencer Feed Ticketing

## Security Assessment (Summary Report)

**July 31, 2026**

*Prepared for:*
**Harry Kalodner, Steven Goldfeder, and Ed Felten**
Offchain Labs

*Prepared by:* **Simone Monica, Nicolas Donboly, and Jaime Iglesias**

# Table of Contents

# Project Summary

## Contact Information

The following project manager was associated with this project:

**Mary O'Brien**, Project Manager
mary.obrien@trailofbits.com

The following engineering director was associated with this project:

**Benjamin Samuels**, Engineering Director, Blockchain
benjamin.samuels@trailofbits.com

The following consultants were associated with this project:

**Nicolas Donboly**, Consultant
nicolas.donboly@trailofbits.com

**Simone Monica**, Consultant
simone.monica@trailofbits.com

**Jaime Iglesias**, Consultant
jaime.iglesias@trailofbits.com

## Project Timeline

The significant events and milestones of the project are listed below.

| Date | Event |
|---|---|
| **July 14, 2026** | Delivery of summary report draft |
| **July 31, 2026** | Delivery of final summary report |

# Project Targets

### Feed-Tickets-Contracts

| | |
|---|---|
| Repository | https://github.com/OffchainLabs/feed-ticket-contracts |
| Version | `9d04eb0be29733b9f302a56976a0f3148f10866e` |
| Type | Solidity |
| Platform | Arbitrum |

# Executive Summary

## Engagement Overview

Offchain Labs engaged Trail of Bits to review the security of their sequencer feed ticketing contracts.

A team of three consultants conducted the review from June 29 to July 1, 2026, for a total of nine engineer-days of effort. With full access to source code and documentation , we performed static and dynamic testing of the target, using automated and manual processes.

## Observations and Impact

This new feature allows users to access a premium sequencer feed by purchasing tickets. The contracts in scope manage all the accounting and purchasing for these tickets.

The main focus of the review was to determine whether this feature was correctly implemented and to identify potential edge cases, missing functionality, and opportunities for improvement. Additionally, we looked for logical issues and Solidity-specific issues.

All findings are enhancements to code quality and relate to undocumented edge cases and UX improvements.

## Recommendations

- **Consider addressing the Code Quality findings presented in appendix A.**

# A. Code Quality Findings

The following findings are not associated with any specific vulnerabilities. However, fixing them will enhance code readability and may prevent the introduction of vulnerabilities in the future.

- **Grandfather guarantee can be broken by lowering `_maxTicketsPerRound` during the grandfather window.** If an admin reduces the `_maxTicketsPerRound`, there could be a situation where some users will not be able to renew their grandfathered tickets because the round does not have enough tickets to cover them.

```
function purchaseTickets(
    uint256 expectedRound,
    uint256 expectedPrice,
    uint256 numTicketsDesired,
    bytes32 apiKeyHash
) external {
    _lazyUpdateRoundState();

    if (numTicketsDesired == 0) revert ZeroTicketsRequested();
    if (expectedRound != _roundNumber) revert RoundNumberMismatch(expectedRound,
_roundNumber);
    if (expectedPrice != _currentPrice) revert IncorrectTicketPrice(expectedPrice,
_currentPrice);
    if (_ticketsSoldThisRound >= _maxTicketsPerRound) {
        revert MaxTicketsSold();
    }

    uint16 _numTicketsDesired = numTicketsDesired.toUint16();
    uint16 _numTickets = uint256(_ticketsSoldThisRound) +
uint256(_numTicketsDesired) > _maxTicketsPerRound
        ? _maxTicketsPerRound - _ticketsSoldThisRound
        : _numTicketsDesired;
    [...]
```

*Figure A.1: part of the `purchaseTickets` function in `Tickets.sol`*

- **Partial fills in `purchaseTickets` are silent.** If there are not enough tickets to cover `numTicketsDesired`, the function will purchase as many as are available.

  However, users have no control over this partial fill; instead, we suggest adding another parameter to the function `_minTicketsDesired` that prevents partial fills unless they cover a minimum number of tickets the user desires.

```
function purchaseTickets(
    uint256 expectedRound,
```

```
    uint256 expectedPrice,
    uint256 numTicketsDesired,
    bytes32 apiKeyHash
) external {
    _lazyUpdateRoundState();

    [...]

    uint16 _numTicketsDesired = numTicketsDesired.toUint16();
    uint16 _numTickets = uint256(_ticketsSoldThisRound) +
uint256(_numTicketsDesired) > _maxTicketsPerRound
        ? _maxTicketsPerRound - _ticketsSoldThisRound
        : _numTicketsDesired;
    uint256 cost = expectedPrice * uint256(_numTickets);

    UserData memory userDataMem = _userData[msg.sender];
    if (userDataMem.tokenBalance < cost) {
        revert InsufficientTokenBalance(userDataMem.tokenBalance, cost);
    }

    [...]
```

*Figure A.2: `_numTickets` clamping in the `purchaseTickets` function in `Tickets.sol`*

# About Trail of Bits

Founded in 2012 and headquartered in New York, Trail of Bits provides technical security assessment and advisory services to some of the world's most targeted organizations. We combine high-end security research with a real-world attacker mentality to reduce risk and fortify code. With 100+ employees around the globe, we've helped secure critical software elements that support billions of end users, including Kubernetes and the Linux kernel.

We maintain an exhaustive list of publications at https://github.com/trailofbits/publications, with links to papers, presentations, public audit reports, and podcast appearances.

In recent years, Trail of Bits consultants have showcased cutting-edge research through presentations at CanSecWest, HCSS, Devcon, Empire Hacking, GrrCon, LangSec, NorthSec, the O'Reilly Security Conference, PyCon, REcon, Security BSides, and SummerCon.

We specialize in software testing and code review assessments, supporting client organizations in the technology, defense, blockchain, and finance industries, as well as government entities. Notable clients include HashiCorp, Google, Microsoft, Western Digital, Uniswap, Solana, Ethereum Foundation, Linux Foundation, and Zoom.

To keep up with our latest news and announcements, please follow @trailofbits on X or LinkedIn and explore our public repositories at https://github.com/trailofbits. To engage us directly, visit our "Contact" page at https://www.trailofbits.com/contact or email us at info@trailofbits.com.

**Trail of Bits, Inc.**
228 Park Ave S #80688
New York, NY 10003
https://www.trailofbits.com
info@trailofbits.com

# Notices and Remarks

## Copyright and Distribution



## Test Coverage Disclaimer

Trail of Bits performed all activities associated with this project in accordance with a statement of work and an agreed-upon project plan.

Security assessment projects are time-boxed and often rely on information provided by a client, its affiliates, or its partners. As a result, the findings documented in this report should not be considered a comprehensive list of security issues, flaws, or defects in the target system or codebase.

Trail of Bits uses automated testing techniques to rapidly test software controls and security properties. These techniques augment our manual security review work, but each has its limitations. For example, a tool may not generate a random edge case that violates a property or may not fully complete its analysis during the allotted time. A project's time and resource constraints also limit their use.