Offchain Labs Arbitrum Quorum Changes

Security Assessment (Summary Report)

January 29, 2026

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

Prepared by: **Jaime Iglesias and Simone Monica**

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:

Jaime Iglesias, Consultant
jaime.iglesias@trailofbits.com

Simone Monica, Consultant
simone.monica@trailofbits.com

Project Timeline

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

Date	Event
November 16, 2025	Delivery of report draft
December 8, 2025	Delivery of initial summary report
January 29, 2026	Delivery of final summary report

Project Targets

The engagement involved reviewing and testing the target listed below.

Nitro

Repository	https://github.com/ArbitrumFoundation/governance
Version	`74e32d4c2baf07918b3367eecb70c1f8786bb79a` `1ee03ed990a1b3c06e6b4e91503c4d245bb42a8f` `a2f690adb9df17f89a810273d3d83a798daca9fb`
Type	Solidity
Platform	EVM

Executive Summary

Engagement Overview

Offchain Labs engaged Trail of Bits to review the security of a governance proposal implementation to alter the quorum calculation mechanism for the Arbitrum DAO, along with its corresponding governance action contract.

A team of two consultants conducted the initial review on November 14, 2025. We also reviewed additional changes introduced after the initial review on January 15, 27, and 28, 2026. 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

The primary targets of the review were the changes implemented in PR #364, which includes modifications to the governance quorum calculation and the corresponding governance action contracts for the upgrade to proceed.

The additional changes for the extended review in January 2026 included those in PR #365, which introduces proposal cancellation and front-running protection, and changes to the L2 governor in PR #371, which adds checkpointing to the quorum settings and modifies the DVP quorum action contract. Note that the actual action contract deployment and payload were not part of the January scope.

Overall, we found the changes to be easy to reason about and the implementation to be clear. We identified only two minor informational-severity findings and some code quality issues.

Recommendations

- **Remediate the findings disclosed in this report.** These findings should be addressed through direct fixes or broader refactoring efforts.

Summary of Findings

The table below summarizes the findings of the review, including details on type and severity.

ID	Title	Type	Severity
1	adjustTotalDelegation allows the new value to be zero	Data Validation	**Informational**
2	L2ArbitrumGovernor does not align with the expected EIP-165 interface	Data Validation	**Informational**

Detailed Findings

1. adjustTotalDelegation allows the new value to be zero

Severity: **Informational**	Difficulty: **High**
Type: Data Validation	Finding ID: TOB-QUO-1
Target: `governance/src/L2ArbitrumToken.sol`	

Description

The `adjustTotalDelegation` function allows the DAO to set the new value to zero.

```
function adjustTotalDelegation(int256 adjustment)
    external
    onlyOwner
{
    uint256 latest = _totalDelegationHistory.latest();
    int256 newValue = int256(latest) + adjustment;

    // negative newValue should be impossible
    // since the adjustment should bring the value to true total delegation
    // which is at minimum zero
    require(newValue >= 0, "ARB: NEGATIVE_TOTAL_DELEGATION");
    _totalDelegationHistory.push(uint256(newValue));

    emit TotalDelegationAdjusted(latest, uint256(newValue));
}
```

Figure 1.1: The `adjustTotalDelegation` function in L2ArbitrumToken.sol#L105–L108

When that happens, the `quorum` function will use the "old way" of computing the quorum, which we find strange.

```
// if pastTotalDelegatedVotes is 0, then blockNumber is almost certainly prior to
the first totalDelegatedVotes checkpoint
// in this case we should use getPastCirculatingSupply to ensure quorum of
pre-existing proposals is unchanged
// in the unlikely event that totalDvp is 0 for a block _after_ the dvp update,
getPastCirculatingSupply will be used with a larger quorumNumerator,
// resulting in a much higher calculated quorum. This is okay because quorum is
clamped.
uint256 calculatedQuorum = (
    (
        pastTotalDelegatedVotes == 0
            ? getPastCirculatingSupply(blockNumber)
```

```
            : pastTotalDelegatedVotes
    ) * quorumNumerator(blockNumber)
) / quorumDenominator();
```

Figure 1.2: The quorum function in L2ArbitrumGovernor.sol#L182–L192

The severity of this finding is informational because `adjustTotalDelegation` is owner-protected; therefore, only the DAO can change the value.

Recommendations

Short term, consider whether this behavior is intended; if it is, then thoroughly document it to inform governance changes. If it is not, then add a check to ensure that `newValue` cannot be zero.

2. L2ArbitrumGovernor does not align with the expected EIP-165 interface

Severity: **Informational**	Difficulty: **Low**
Type: Data Validation	Finding ID: TOB-QUO-2
Target: `governance/src/L2ArbitrumGovernor.sol`	

Description

The `L2ArbitrumGovernor` contract follows EIP-165, which provides a mechanism for smart contracts to inform others of which interface they implement. This is done through the `supportsInterface` function, which, as shown in figure 2.1, makes an internal call to the same function of a parent contract.

```
function supportsInterface(bytes4 interfaceId)
    public
    view
    override(GovernorUpgradeable, GovernorTimelockControlUpgradeable)
    returns (bool)
{
    return GovernorTimelockControlUpgradeable.supportsInterface(interfaceId);
}
```

Figure 2.1: The `supportsInterface` function in `L2ArbitrumGovernor.sol#L340`

PR #365 introduces a new function, `cancel`. This function is not part of the public interfaces of any of `L2ArbitrumGovernor`'s parent contracts, which means it cannot communicate its public interface effectively.

Recommendations

Short term, update the `supportsInterface` function of `L2ArbitrumGovernor` to reflect the new interface ID being supported.

Long term, whenever changes are made to the public interface of a contract that implements EIP-165, make sure they are properly updated.

A. Vulnerability Categories

The following tables describe the vulnerability categories, severity levels, and difficulty levels used in this document.

Vulnerability Categories	
Category	**Description**
Access Controls	Insufficient authorization or assessment of rights
Auditing and Logging	Insufficient auditing of actions or logging of problems
Authentication	Improper identification of users
Configuration	Misconfigured servers, devices, or software components
Cryptography	A breach of system confidentiality or integrity
Data Exposure	Exposure of sensitive information
Data Validation	Improper reliance on the structure or values of data
Denial of Service	A system failure with an availability impact
Error Reporting	Insecure or insufficient reporting of error conditions
Patching	Use of an outdated software package or library
Session Management	Improper identification of authenticated users
Testing	Insufficient test methodology or test coverage
Timing	Race conditions or other order-of-operations flaws
Undefined Behavior	Undefined behavior triggered within the system

Severity Levels	
Severity	**Description**
Informational	The issue does not pose an immediate risk but is relevant to security best practices.
Undetermined	The extent of the risk was not determined during this engagement.
Low	The risk is small or is not one the client has indicated is important.
Medium	User information is at risk; exploitation could pose reputational, legal, or moderate financial risks.
High	The flaw could affect numerous users and have serious reputational, legal, or financial implications.

Difficulty Levels	
Difficulty	**Description**
Undetermined	The difficulty of exploitation was not determined during this engagement.
Low	The flaw is well known; public tools for its exploitation exist or can be scripted.
Medium	An attacker must write an exploit or will need in-depth knowledge of the system.
High	An attacker must have privileged access to the system, may need to know complex technical details, or must discover other weaknesses to exploit this issue.

B. 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.

- The `setQuorumMinxAndMax` function does not emit an event.

```
function setQuorumMinAndMax(uint256 _minimumQuorum, uint256 _maximumQuorum)
    external
    onlyGovernance
{
    require(_minimumQuorum < _maximumQuorum, "L2ArbitrumGovernor:
MIN_GT_MAX");
    minimumQuorum = _minimumQuorum;
    maximumQuorum = _maximumQuorum;
}
```

Figure B.1: The `setQuorumMinxAndMax` function in L2ArbitrumGovernor.sol#L138–L145

- It may not be necessary to use the `virtual` modifier in `_delegate` for the most derived contract necessary. Consider whether the `virtual` keyword can be removed.

```
/// @dev Override ERC20VotesUpgradeable to update total delegation history
when delegation changes
function _delegate(address delegator, address delegatee) internal virtual
override {
    _updateDelegationHistory(delegates(delegator), delegatee,
balanceOf(delegator));
    super._delegate(delegator, delegatee);
}
```

Figure B.2: The `_delegate` function in L2ArbitrumToken.sol#L169–L173

- The NatSpec comments in the DVP quorum action contract need to be updated to reflect the new behavior: there are only six actions now, not eight.

```
    /// @notice Performs the following:
    ///         1. Upgrades the token contract
    ///         2. Calls postUpgradeInit on the token contract to set the
initial total delegation estimate
    ///         3. Upgrades the core governor contract
    ///         4. Sets the new quorum numerator for the core governor
    ///         5. Sets the quorum min/max for the core governor
    ///         6. Upgrades the treasury governor contract
    ///         7. Sets the new quorum numerator for the treasury governor
    ///         8. Sets the quorum min/max for the treasury governor
```

Figure B.3: The NatSpec comments of the `perform` function in ActivateDVPQuorumAction.sol#L63–L71

- The documentation does not indicate that the actual maximum value the quorum can take is `uint224.max`, even though the value being passed is a `uint256`. This is because the checkpointing library only supports `uint224`. Update the documentation to include this information.

```
function push(History storage self, uint256 value) internal returns (uint256,
uint256) {
    uint256 pos = self._checkpoints.length;
    uint256 old = latest(self);
    if (pos > 0 && self._checkpoints[pos - 1]._blockNumber == block.number) {
        self._checkpoints[pos - 1]._value =
SafeCastUpgradeable.toUint224(value);
```

Figure B.4: The `push` function in CheckpointsUpgradeable.sol#L60–L64

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.