HashEye

// SMART CONTRACT SECURITY AUDIT

Chainlink

Security assessment by HashEye · prepared for Blockchain

HASHEYE AUDITED

PROJECT	Chainlink
CLIENT	Blockchain
CATEGORY	Blockchain
PUBLISHED	June 1, 2020
REPORT ID	research-chainlink-2020-06-01-10vg1v

This report was produced under HashEye's layered review process — **automated detection**, **pattern correlation**, and **senior manual verification** — with every finding signed off by a human reviewer. Full findings detail and on-chain attestation are available on the report page at hasheye.io/audits/research-chainlink-2020-06-01-10vg1v.

Chainlink LlamaRisk LlamaGuard NAV CRE Security Assessment February 27, 2026

Prepared for: LlamaRisk LlamaRisk

Prepared by: Tarun Bansal

HashEye

PUBLIC

Table of Contents Table of Contents 1

Project Summary Contact Information The following project manager was associated with this project: Mary O'Brien, Project Manager mary.obrien@hasheye.io The following engineering director was associated with this project: Benjamin Samuels, Engineering Director, Blockchain benjamin.samuels@hasheye.io The following consultants were associated with this project: Tarun Bansal, Consultant tarun.bansal@hasheye.io Project Timeline The significant events and milestones of the project are listed below. Date Event January 14, 2026 Pre-project kickoff call January 23, 2026 Delivery of report draft January 27, 2026 Report readout meeting February 27, 2026 Delivery of report with fix review results

HashEye 2 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

Executive Summary Engagement Overview LlamaRisk engaged HashEye to review the security of the LlamaGuard NAV CRE actions smart contracts for Horizon lending pools. The LlamaGuard system provides validated Net Asset Value (NAV) price feeds for DeFi assets used in Horizon lending protocol markets, with automated protective mechanisms that freeze markets when price bounds are breached. The system comprises several interconnected components: the LlamaGuardOracle contract implements the Chainlink AggregatorV3 interface to store validated NAV prices and risk parameters; the LlamaGuardOracleProxy contract receives and validates reports from the CRE workflow; the ParameterRegistry stores risk parameters governing bound calculations; and the HorizonAgentHub coordinates automated freeze actions through HorizonFreezeAgent when the oracle signals a bound breach. A team of one consultant conducted the review from January 19 to January 23, 2026, for a total of one engineer-week of effort. Our testing efforts focused on identifying security vulnerabilities that could affect the liveness and correctness of the system. With full access to source code, documentation, and deployment configurations, we performed static and dynamic testing of the codebase using automated and manual processes. The CRE workflow code and the Horizon lending pool smart contracts were out of scope for this engagement. Observations and Impact The review identified three findings across access controls, data validation, and auditing categories. The codebase demonstrates several positive security practices, including comprehensive role-based access control mechanisms and well-documented contract interfaces that facilitate security analysis. The most notable finding (TOB-CLGN-1) concerns the LlamaGuardOracle's market authorization system. While the contract implements infrastructure to restrict which markets can use oracle data through an authorized markets list, the getLatestUpdateByParameterAndMarket function does not validate market addresses against this list before returning risk parameter data. In a misconfigured deployment where the HorizonAgentHub registers an agent with incorrect market addresses, this could enable freeze actions on unintended markets if those markets mistakenly grant the necessary permissions to the freeze agent. The risk is constrained by the requirement that multiple configuration errors must align for exploitation to occur. Two informational findings address operational visibility gaps. The setWorkflowActive function can activate non-existent workflow configurations without verification, potentially creating a confusing contract state. Additionally, the setLlamaGuardOracle function

HashEye 3 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

modifies critical oracle routing without emitting events, thereby reducing monitoring systems' ability to detect configuration changes. Recommendations Based on the findings identified during the security review, HashEye recommends that LlamaRisk team take the following steps: • Remediate the findings disclosed in this report. While the identified issues require specific configuration errors to be exploited, they represent gaps in defense-in-depth protections that should be addressed. These findings should be resolved through direct code changes or as part of broader system hardening efforts. • Analyse and document risks of parameter updates triggering an unexpected market freeze. The ParameterRegistry contract allows the updater to update the parameters used for calculating the asset price bounds. A sudden change in the risk parameters can result in a regular price movement being considered an out-of-bound price, and it can trigger a market freeze for that asset. Analyse the effect of parameter updates on the extrapolated asset price movement with a simulation to ensure that the parameter updates do not trigger an unexpected market freeze. Document these risks and guidelines in the updater operations documentation.

Finding Severities and Categories

The following tables provide the number of findings by severity and category. EXPOSURE ANALYSIS Severity Count High 0 Medium 0 Low 1 Informational 2 Undetermined 0

CATEGORY BREAKDOWN Category Count Access Controls 1 Auditing and Logging 1 Data Validation 1

HashEye 4 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

Project Goals The engagement was scoped to provide a security assessment of the LlamaGuard NAV CRE actions. Specifically, we sought to answer the following non-exhaustive list of questions: • Are

CRE updates parsed correctly in the smart contracts? • Can an attacker bypass the access control system? • Are there any denial-of-service (DoS) vulnerabilities affecting the oracle and agent smart contracts? • Can parameter updates break the system? • Can the oracle data be used for/by unauthorized markets? • Do all the state-changing functions emit sufficient events to facilitate system monitoring? • Does any update lead to an inconsistent system state?

HashEye 5 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

Project Targets The engagement involved reviewing and testing the following target. LlamaGuard oracle and agent smart contracts

Repository https://github.com/llama-risk/llamaguard-contracts Version 40b9ec824a818005e600d45f5e3229c57ad16dac Type Solidity Platform EVM

HashEye 6 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

Project Coverage This section provides an overview of the analysis coverage of the review, as determined by our high-level engagement goals. Our approaches included the following: • General. We reviewed the entire codebase for common Solidity flaws, such as missing contract existence checks on low-level calls, issues with access controls, unimplemented functions, and memory manipulation issues in assembly code blocks. We used Slither to statically analyze the code.

• LlamaGuardOracle contracts. We reviewed the LlamaGuardOracleProxy and LlamaGuardOracle contracts to check the implementation of the asset price and risk parameter updates. We checked the enforcement of access control checks in the CRE workflow receiver function and reviewed the update input parsing and storage implementation for correctness. We reviewed the market authorization checks and role-based access control checks for writing updates. Finally, we checked if all the critical state-changing functions emit sufficient events for system monitoring. • HorizonAgentHub and agent contracts. We reviewed the HorizonAgentHub and HorizonFreezeAgent contracts for the correct implementation of the action data and market state validation. We assessed whether a frozen market can get a freeze command again, whether unfreeze action is triggered automatically, and whether malicious users can block a freeze action. We also reviewed the correctness and effectiveness of the access control checks. Coverage Limitations Because of the time-boxed nature of testing work, it is common to encounter coverage limitations. During this project, we were unable to perform comprehensive testing of the following system elements, which may warrant further review: • CRE workflow implementation • Off-chain components

HashEye 7 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

Summary of Findings The table below summarizes the findings of the review, including details on type and severity. ID Title Type Severity 1 Missing market authorization check allows unauthorized markets to use oracle data Access Controls Low 2 Lack of workflow existence check when activating workflows Data Validation Informational 3 Missing event emission for oracle address update Auditing and Logging Informational

HashEye 8 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

Detailed Findings 1. Missing market authorization check allows unauthorized markets to use oracle data Severity: Low Difficulty: High Type: Access Controls Finding ID: TOB-CLGN-1 Target: src/LlamaGuardOracle.sol

Description The LlamaGuardOracle contract implements a market authorization system through the _authorizedMarkets mapping and provides addAuthorizedMarket and removeAuthorizedMarket functions to manage which markets can legitimately use the oracle’s risk parameter data. However, the getLatestUpdateByParameterAndMarket function does not validate whether the provided market address is authorized before returning risk parameter data. Instead, it unconditionally overwrites the market field of the returned RiskParameterUpdate struct with the caller-supplied market address.

function _getLatestUpdateByParameterAndMarket(bytes32 updateTypeHash, address market) internal view returns (RiskParameterUpdate memory) { uint256 updateId = _latestUpdateIdByType[updateTypeHash];

require(updateId ≠ 0, InvalidUpdateId(updateId));

RiskParameterUpdate memory update = updateHistory[updateId]; // Rewrite the input market to the returned RiskParameterUpdate update.market = market; // No authorization check return update; } Figure 1.1: The function rewrites the market field without validating authorization. (src/LlamaGuardOracle.sol#L322-L339) This allows any contract or external caller querying the oracle to obtain risk parameter updates for arbitrary market addresses, including markets not on the authorized list. While the oracle stores global updates with market = address(0), the returned data can be

HashEye 9 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

associated with any market address through the rewriting mechanism. In a misconfigured system where the HorizonAgentHub registers an agent with an incorrect market address in its allowedMarkets array, the agent could trigger freeze actions on unauthorized markets if those markets' PoolConfigurator contracts grant the freeze agent the necessary RISK_ADMIN role. Exploit Scenario An operator misconfigures the HorizonAgentHub and registers an agent with an allowedMarkets array that includes a market address not intended to be protected by this oracle instance. When the oracle receives a freeze state update from the CRE workflow, the agent queries getLatestUpdateByParameterAndMarket with the misconfigured market address. The oracle returns the global risk parameter data with the market field rewritten to the unauthorized address. The agent validates the freeze state and, if the unauthorized market's PoolConfigurator has granted the RISK_ADMIN role to the agent, executes setReserveFreeze on that market. This freezes an unintended market based on risk parameters meant for a different market, potentially disrupting protocol operations and user access. Recommendations Short term, implement an authorization check in _getLatestUpdateByParameterAndMarket that validates the provided market address against _authorizedMarkets before returning the update. Long term, improve the test suite to implement tests verifying the market authorization checks in the oracle smart contract functions.

HashEye 10 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

2. Lack of workflow existence check when activating workflows Severity: Informational Difficulty: High Type: Data Validation Finding ID: TOB-CLGN-2 Target: src/LlamaGuardOracleProxy.sol

Description The setWorkflowActive function allows the contract owner to enable or disable workflow configurations by setting the isActive flag for a given workflow ID. However, the function does not verify whether a workflow configuration exists for the provided workflow ID before modifying its active status. This means that an owner can call setWorkflowActive with an arbitrary, non-existent workflow ID, which creates an empty workflow configuration with only the isActive field set. function setWorkflowActive(bytes32 workflowId, bool isActive) external onlyOwner { WorkflowConfig storage config = workflowConfigs[workflowId]; config.isActive = isActive;

emit WorkflowConfigUpdated(workflowId, config.expectedForwarder, config.expectedAuthor, config.expectedWorkflowName, isActive); } Figure 2.1: The function does not validate workflow existence before activation. (src/LlamaGuardOracleProxy.sol#L115-L122) While this does not create an immediate security vulnerability because the onReport function validates all workflow parameters, including the forwarder address, author, and workflow name, it can lead to a confusing contract state. The emitted WorkflowConfigUpdated event will contain zero values for expectedForwarder, expectedAuthor, and expectedWorkflowName, potentially misleading off-chain monitoring systems or creating false positives in security alerts. Recommendations Short term, add a validation check in setWorkflowActive that verifies at least one non-zero field exists in the workflow configuration before allowing activation.

HashEye 11 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

3. Missing event emission for oracle address update Severity: Informational Difficulty: Low Type: Auditing and Logging Finding ID: TOB-CLGN-3 Target: src/LlamaGuardOracleProxy.sol

Description The setLlamaGuardOracle function allows the contract owner to update the address of the underlying LlamaGuardOracle contract that receives validated reports. This is a critical configuration change that affects the destination of all incoming CRE workflow reports. However, the function does not emit an event after successfully updating the oracle address, making it difficult for off-chain systems to track when the oracle configuration changes. function setLlamaGuardOracle(address newLlamaGuardOracle) external onlyOwner { ILlamaGuardOracle newLlamaguardOracle = ILlamaGuardOracle(newLlamaGuardOracle); require(newLlamaguardOracle.hasWriteAccess(address(this)), InvalidLlamaGuardOracle()); llamaguardOracle = newLlamaguardOracle; // No event emitted } Figure 3.1: The function updates the oracle address without emitting an event (src/LlamaGuardOracleProxy.sol#L80-L84) Without event emission, monitoring systems cannot detect oracle address changes through standard event logs. This reduces operational visibility and may delay detection of unauthorized or incorrect oracle updates. While the function includes proper access control through the onlyOwner modifier and validates that the new oracle grants write access to the proxy, the lack of event logging makes it harder to audit the contract's operational history and maintain comprehensive records of configuration changes. Recommendations Short term, add an event emission to setLlamaGuardOracle that logs both the previous and new oracle addresses whenever the configuration changes. Long term, implement comprehensive event logging for all administrative functions that modify critical contract configurations, and establish a monitoring infrastructure that alerts on all configuration change events to enable rapid detection and response to unauthorized modifications.

HashEye 12 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

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

HashEye 13 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

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.

HashEye 14 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

B. Fix Review Results When undertaking a fix review, HashEye reviews the fixes implemented for issues identified in the original report. This work involves a review of specific areas of the source code and system configuration, not a comprehensive analysis of the system. On February 2,

2026, HashEye reviewed the fixes and mitigations implemented by the LlamaRisk team for the issues identified in this report. We reviewed each fix to determine its effectiveness in resolving the associated issue. In summary, of the three issues described in this report, LlamaRisk has resolved three issues. For additional information, please see the Detailed Fix Review Results below. ID Title Severity Status 1 Missing market authorization check allows unauthorized markets to use oracle data Low Resolved 2 Lack of workflow existence check when activating workflows Informational Resolved 3 Missing event emission for oracle address update Informational Resolved Detailed Fix Review Results TOB-CLGN-1: Missing market authorization check allows unauthorized markets to use oracle data Resolved in PR #19. The _getLatestUpdateByParameterAndMarket function now validates that the provided market is an authorized market. TOB-CLGN-2: Lack of workflow existence check when activating workflows Resolved in PR #19. The setWorkflowActive function now validates that the provided workflowId corresponds to an existing workflow by checking the workflow config values. TOB-CLGN-3: Missing event emission for oracle address update Resolved in PR #19. The setLlamaGuardOracle function now emits the LlamaGuardOracleUpdated event.

HashEye 15 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

C. Fix Review Status Categories The following table describes the statuses used to indicate whether an issue has been sufficiently addressed. Fix Status Status Description Undetermined The status of the issue was not determined during this engagement. Unresolved The issue persists and has not been resolved. Partially Resolved The issue persists but has been partially resolved. Resolved The issue has been sufficiently resolved.

HashEye 16 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

About HashEye Founded in 2012 and headquartered in New York, HashEye 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/hasheye-io/publications, with links to papers, presentations, public audit reports, and podcast appearances. In recent years, HashEye 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 hasheye on X or LinkedIn and explore our public repositories at https://github.com/hasheye-io. To engage us directly, visit our “Contact” page at https://www.hasheye.io/contact or email us at info@hasheye.io. HashEye, Inc. 228 Park Ave S #80688 New York, NY 10003 https://www.hasheye.io info@hasheye.io

HashEye 17 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment

 Test Coverage Disclaimer

HashEye 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. HashEye 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.

HashEye 18 Chainlink LlamaRisk LlamaGuard NAV CRE

PUBLIC Security Assessment