ANASTASIA LABS

Proof of Achievement - Milestone 1

OpShin Audit

Project Number	1200175
Project Manager	Jonathan Rodriguez

Contents

Project Name: OpShin Audit
URL: Catalyst Proposal

Project Goal

The primary goal of the OpShin audit project is to enhance the reliability and security of smart contracts developed using the OpShin language within the Cardano ecosystem. This is achieved through a comprehensive audit that identifies vulnerabilities, addresses edge cases, and optimizes the language's efficiency. By emphasizing transparency and collaboration, the project aims to support developers with detailed documentation and best practices, ultimately elevating the quality of smart contracts. The anticipated outcomes include a significant reduction in reported vulnerabilities and the establishment of a robust foundation for safe and trustworthy smart contract development on Cardano using OpShin.

Project Deliverables

OpShin Language Analysis

- **Deliverable**: **Detailed Analysis Report**
  - **Description**: The audit team will produce a comprehensive report that identifies vulnerabilities and areas for improvement within the OpShin language codebase.
  - **Key Activities**:
    - Utilize automated analysis tools to support findings.
    - Conduct unit tests to evaluate the functionality of the code.
    - Perform manual assessments to complement automated analyses.
  - **Outcome**: This report will serve as a foundational document for understanding the current state of the OpShin language and guiding future enhancements.

Edge Case Identification

- **Deliverable**: **Documented List of Edge Cases**
  - **Description**: The audit team will compile a thorough list of edge cases relevant to the development of smart contracts using OpShin.
  - **Key Activities**:
    - Identify and document edge cases through extensive manual analysis.
    - Propose strategies for addressing these edge cases that can be applied in future iterations of the language.
    - Utilize tailored analysis tools to support the identification process.
  - **Outcome**: This documentation will provide valuable insights into potential pitfalls in smart contract development, helping developers avoid common errors.

Draft Audit Report Preparation and Feedback Integration

- **Deliverable**: **Comprehensive Audit Report**
  - **Description**: The audit team will compile a detailed audit report that outlines identified vulnerabilities, recommended fixes, and best practices for the development of OpShin.
  - **Key Activities**:
    - Document vulnerabilities and provide actionable recommendations.
    - Collaborate with the OpShin team to integrate feedback and address reported issues in the codebase.
    - Ensure that the audit report reflects the most current state of the code following these integrations.
  - **Outcome**: This report will serve as a crucial resource for developers, guiding them in improving the reliability and security of smart contracts written in OpShin.

Public Dissemination and Resolution of Remaining Issues

- **Deliverable**: **Finalized Audit Report and Presentation**
  - **Description**: The finalized audit report will be publicly shared, and the findings will be presented to the Cardano community.
  - **Key Activities**:
    - Disseminate the audit report through appropriate channels to ensure community access.
    - Present findings and recommendations in a format that encourages discussion and further inquiry.
    - Address remaining medium and low-priority findings from the report, ensuring all issues are resolved.
    - Produce a final close-out report summarizing the project outcomes and lessons learned.
    - Create a final close-out video to visually represent the project's achievements and key takeaways.

- **Outcome**: This will enhance community trust and engagement, providing transparency into the auditing process and supporting ongoing improvements in the OpShin language.

Project Timelines

Signatures

Audit Objectives

1. Comprehensive Audit of OpShin Language: Conduct a thorough audit of the OpShin language used for smart contract development, ensuring meticulous scrutiny of the codebase to identify vulnerabilities and inefficiencies.
2. Address Edge Cases and Optimize Efficiency: Identify and address edge cases within the OpShin language to enhance the efficiency and reliability of smart contracts, thereby safeguarding user assets in the Cardano ecosystem.
3. Engagement of Experienced Professionals: Collaborate with developers and auditors who have expertise in smart contract development to ensure a robust and informed auditing process.
4. Facilitate the Auditing Process: Provide detailed documentation that supports the auditing process, ensuring comprehensive coverage of potential edge cases and vulnerabilities.
5. Promote Transparency and Collaboration: Emphasize transparency and collaboration throughout the audit, leveraging the open-source nature of OpShin to foster trust and community involvement.
6. Enhance Quality and Security: Elevate the quality of smart contracts written in OpShin, reinforcing Cardano's reputation as a secure and trustworthy blockchain platform.

Specific Area of Focus

Our manual code auditing is focused on a wide range of attack vectors, including but not limited to:

1. Language Structure and Syntax
   - Evaluate the basic syntax and structure of the language
   - Check for consistency and clarity in language design
2. Security Features
   - Assess built-in security mechanisms
   - Identify potential vulnerabilities in the language design
3. Performance and Efficiency
   - Analyze the language's runtime performance
   - Evaluate memory management and resource utilization
4. Compatibility and Interoperability
   - Check compatibility with other languages and systems
   - Assess ease of integration with existing codebases
5. Standard Library and Ecosystem
   - Evaluate the completeness and quality of the standard library
   - Assess the availability and quality of third-party libraries

6. Error Handling and Debugging
   - Review error handling mechanisms
   - Assess debugging tools and capabilities

7. Scalability
   - Evaluate the language's ability to handle large-scale projects
   - Assess support for concurrent and parallel programming

8. Documentation and Learning Resources
   - Review the quality and completeness of official documentation
   - Assess the availability of learning resources and community support

9. Testing and Quality Assurance
   - Evaluate built-in testing frameworks
   - Assess tools for code quality and static analysis

10. Compliance and Standards
    - Verify adherence to relevant industry standards
    - Check compliance with regulatory requirements, if applicable

11. Vulnerabilities and Threat Modeling
    - Identify potential security threats specific to the language
    - Assess the language's resilience against common attack vectors

Audit Timeline

- **Weeks 1-2**: Discovery and Planning
  - Familiarization with the codebase and project specifications.
- **Weeks 3-8**: Manual Review
  - Perform an in-depth review of the code to identify vulnerabilities.
  - Publish the initial findings report.
  - Collaborate with the Client to implement suggested fixes.
  - Produce a finalized audit report.

Approvals

Operational Communication Channels

Communication Channels

Participation Evidence