Synthetix Menkalinan Release Smart Contract Audit

1. Introduction

iosiro was commissioned by Synthetix to conduct a smart contract audit of their Alkaid Release, which included the following component:

- SIP-167 from 2 to 4 August and 20 to 23 August with one auditor, consuming a total of 5 resource days.

This report is organized into the following sections.

- **Section 2 - Executive summary:** A high-level description of the findings of the audit.
- **Section 3 - Audit details:** A description of the scope and methodology of the audit.

- **Section 4 - Design specification:** An outline of the intended functionality of the smart contracts.
- **Section 5 - Detailed findings:** Detailed descriptions of the findings of the audit.

The information in this report should be used to understand the smart contracts' risk exposure better and as a guide to improving the security posture of the smart contracts by remediating issues identified. The results of this audit reflect the in-scope source code reviewed at the time of the audit.

The purpose of this audit was to achieve the following:

- Identify potential security flaws.
- Ensure that the smart contracts function according to the documentation provided.

Assessing the off-chain functionality associated with the contracts, for example, backend web application code, was outside of the scope of this audit.

Due to the unregulated nature and ease of transfer of cryptocurrencies, operations that store or interact with these assets are considered high risk from cyber attacks. As such, the highest level of security should be observed when interacting with these assets. This requires a forward-thinking approach, which takes into account the new and experimental nature of blockchain technologies. Strategies that should be used to encourage secure code development include:

- Security should be integrated into the development lifecycle, and the level of perceived security should not be limited to a single code audit.
- Defensive programming should be employed to account for unforeseen circumstances.
- Current best practices should be followed where possible.

2. Executive summary

This report presents the findings of a smart contract audit performed by iosiro of Synthetix's Menkalinan release.

SIP-167 introduced a bridge to connect L2 contracts to a Gnosis Safe on L1, in order to facilitate governance actions through the ProtocolDAO.

No security issues were found in the code. Two code style suggestions were made and implemented by Synthetix.

The following additional points were noted during the audit:

- Tests showed that the order of transactions were consistent between L1 and L2 and that relayed messages are deduplicated by the Messenger. However, the introduction of a nonce would provide additional assurance that transaction order is maintained and no duplicate relay messages are processed. This would also mitigate the small chance of a race between `directRelay` calls and `finalizeRelay` messages. This would be a defence in depth measure and the added complexity most likely out weighs the benefit.
- Publishing snippets would make it easier to decode the events, particularly `RelayBatchInitiated` and `RelayBatchFinalized`.
- The contracts were tested with the assumption that `expiryTime` will be 60 days, as per the unit tests and default configuration. iosiro recommends having a deployment strategy for extending this expiry time in case of emergency. An example strategy could be to deploy a new `OwnerRelayOnEthereum`, migrate ownership and then update L1 `CONTRACT_OVM_OWNER_RELAY_ON_OPTIMISM`.

3. Audit details

3.1 Scope

The source code considered in-scope for the assessment is described below. Code from all other files was considered to be out-of-scope. Out-of-scope code that interacts with in-scope code was assumed to function as intended and not introduce any functional or security vulnerabilities for the purposes of this audit.

3.1.1 Synthetix SIP-167 smart contracts

SIP-167

Commit: b888989
Review commits: f16795c, 115701d
Files: MixinSystemSettings.sol, OwnerRelayOnEthereum.sol, OwnerRelayOnOptimism.sol, SynthetixBridgeEscrow.sol, TemporarilyOwned.sol

3.2 Methodology

A variety of techniques were used in order to perform the audit. These techniques are briefly described below.

3.2.1 Code review

The source code was manually inspected to identify potential security flaws. Code review is a useful approach for detecting security flaws, discrepancies between the specification and implementation, design improvements, and high-risk areas of the system.

3.2.2 Dynamic analysis

The contracts were compiled, deployed, and tested in a test environment, both manually and through the test suite provided. Manual analysis was used to confirm that the code was functional and discover security issues that could be exploited.

3.2.3 Automated analysis

Tools were used to automatically detect the presence of several types of security vulnerabilities, including reentrancy, timestamp dependency bugs, and transaction-ordering dependency bugs. Static analysis results were reviewed manually and any false positives were removed. Any true positive results are included in this report.

Static analysis tools commonly used include Slither, Securify, and MythX. Tools such as the Remix IDE, compilation output, and linters could also be used to identify potential areas of concern.

3.3 Risk ratings

Each issue identified during the audit has been assigned a risk rating. The rating is determined based on the criteria outlined below.

- **High risk** - The issue could result in a loss of funds for the contract owner or system users.
- **Medium risk** - The issue resulted in the code specification being implemented incorrectly.
- **Low risk** - A best practice or design issue that could affect the security of the contract.
- **Informational** - A lapse in best practice or a suboptimal design pattern that has a minimal risk of affecting the security of the contract.
- **Closed** - The issue was identified during the audit and has since been satisfactorily addressed, removing the risk it posed.

4. Design specification

The following section outlines the intended functionality of the system at a high level.

4.1 SIP-120

The specification of SIP-167 was based on commit hash c2fe5f3.

5. Detailed findings

The following section includes in-depth descriptions of the findings of the audit.

5.1 High risk

No high-risk issues identified during the audit were present at the conclusion of the audit.

5.2 Medium risk

No medium-risk issues identified during the audit were present at the conclusion of the audit.

5.3 Low risk

No low-risk issues identified during the audit were present at the conclusion of the audit.

5.4 Informational

No identified informational issues were present at the conclusion of the audit.

5.5 Closed

5.5.1 Design comments (informational)

Actions to improve the functionality and readability of the codebase are outlined below.

Naming conventions

OwnerRelayOnEthereum.sol#37

The function name `_getxGasLimit` does not adhere to Solidity naming conventions. iosiro recommends `_getCrossDomainMessageGasLimit` or `_getCrossDomainGasLimit`.

Update

Changed to `_getCrossDomainGasLimit` in f16795c.

Use of `now`

TemporarilyOwned.sol#11,20

In Solidity version 0.7.0, the `now` keyword was deprecated. Developers are encouraged to use `block.timestamp` instead, to ensure forward compatibility of the source code.

Update

Fixed in f16795c.