Audit Report

Snowbridge Updates

v1.0

August 16, 2024

Table of Contents

License

Disclaimer

THE CONTENT OF THIS AUDIT REPORT IS PROVIDED “AS IS”, WITHOUT REPRESENTATIONS AND WARRANTIES OF ANY KIND.

THE AUTHOR AND HIS EMPLOYER DISCLAIM ANY LIABILITY FOR DAMAGE ARISING OUT OF, OR IN CONNECTION WITH, THIS AUDIT REPORT.

THIS AUDIT REPORT WAS PREPARED EXCLUSIVELY FOR AND IN THE INTEREST OF THE CLIENT AND SHALL NOT CONSTRUE ANY LEGAL RELATIONSHIP TOWARDS THIRD PARTIES. IN PARTICULAR, THE AUTHOR AND HIS EMPLOYER UNDERTAKE NO LIABILITY OR RESPONSIBILITY TOWARDS THIRD PARTIES AND PROVIDE NO WARRANTIES REGARDING THE FACTUAL ACCURACY OR COMPLETENESS OF THE AUDIT REPORT.

FOR THE AVOIDANCE OF DOUBT, NOTHING CONTAINED IN THIS AUDIT REPORT SHALL BE CONSTRUED TO IMPOSE ADDITIONAL OBLIGATIONS ON COMPANY, INCLUDING WITHOUT LIMITATION WARRANTIES OR LIABILITIES.

This audit has been performed by

Oak Security GmbH

https://oaksecurity.io/
info@oaksecurity.io

Introduction

Purpose of This Report

Oak Security has been engaged by Snowfork to perform a security audit of updates to Snowbridge.

The objectives of the audit are as follows:

1. Determine the correct functioning of the protocol, in accordance with the project specification.
2. Determine possible vulnerabilities, which could be exploited by an attacker.
3. Determine smart contract bugs, which might lead to unexpected behavior.
4. Analyze whether best practices have been applied during development.
5. Make recommendations to improve code safety and readability.

This report represents a summary of the findings.

As with any code audit, there is a limit to which vulnerabilities can be found, and unexpected execution paths may still be possible. The author of this report does not guarantee complete coverage (see disclaimer).

Codebase Submitted for the Audit

The audit has been performed on the following targets:

Repository	https://github.com/Snowfork/snowbridge
Commit	`10875e90cd1cf35cb039a382586f767ba19a8b9e`

Scope	The scope was restricted to the changes compared to commit `1e27bce2c34989fd48f77d0ac8b6909ff09793c7` for the following directories and files:
	
	`.`
	`└── contracts`

Repository	https://github.com/paritytech/polkadot-sdk
Commit	`b42d1968d93e1f22d402aa831a28ad1545b3d71a`

Scope	The scope was restricted to the changes compared to commit `1e27bce2c34989fd48f77d0ac8b6909ff09793c7` of the https://github.com/Snowfork/snowbridge repository for the following directories and files:
	
	`.`
	`└── bridges`
	`    └── snowbridge`
	`        ├── pallets`
	`        ├── primitives`
	`        └── runtime`
	`            └── runtime-common`

Repository	https://github.com/polkadot-fellows/runtimes
Commit	`7a523172a2c63eab94356ef0bfb2e4c375efc724`

Scope	The scope was restricted to the modifications related to Snowbridge of the following directories and files:
	
	`.`
	`└── system-parachains`
	`    ├── asset-hubs`
	`    │   ├── asset-hub-kusama`
	`    │   └── asset-hub-polkadot`
	`    └── bridge-hubs`
	`        ├── bridge-hub-kusama`
	`        └── bridge-hub-polkadot`

Methodology

The audit has been performed in the following steps:

1. Gaining an understanding of the code base's intended purpose by reading the available documentation.
2. Automated source code and dependency analysis.
3. Manual line-by-line analysis of the source code for security vulnerabilities and use of best practice guidelines, including but not limited to:
    a. Race condition analysis
    b. Under-/overflow issues
    c. Key management vulnerabilities
4. Report preparation

Functionality Overview

Snowbridge is a general-purpose, trustless, and decentralized bridge between Polkadot and Ethereum. This is achieved by using light clients. The protocol uses a BEEFY light client implemented in Solidity smart contracts to track the Polkadot chain, and an Altair-compliant light client to keep track of the Ethereum Beacon Chain implemented in a Substrate pallet.

How to Read This Report

This report classifies the issues found into the following severity categories:

Severity	Description
Critical	A serious and exploitable vulnerability that can lead to loss of funds, unrecoverable locked funds, or catastrophic denial of service.
Major	A vulnerability or bug that can affect the correct functioning of the system, lead to incorrect states or denial of service.
Minor	A violation of common best practices or incorrect usage of primitives, which may not currently have a major impact on security, but may do so in the future or introduce inefficiencies.
Informational	Comments and recommendations of design decisions or potential optimizations, that are not relevant to security. Their application may improve aspects, such as user experience or readability, but is not strictly necessary. This category may also include opinionated recommendations that the project team might not share.

The status of an issue can be one of the following: **Pending,** **Acknowledged**, or **Resolved**.

Note that audits are an important step to improving the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of the system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**. We include a table with these criteria below.

Note that high complexity or low test coverage does not necessarily equate to a higher risk, although certain bugs are more easily detected in unit testing than in a security audit and vice versa.

Code Quality Criteria

The auditor team assesses the codebase's code quality criteria as follows:

Criteria	Status	Comment
Code complexity	**High**	The code implements complex operations and makes use of the latest features coming from Substrate and Cumulus. It also uses the latest XCM specification. The bridge uses/integrates with low-level functionality from different ecosystems.
Code readability and clarity	**Medium**	-
Level of documentation	**High**	The protocol and the modifications applied for Snowbridge to runtimes are well documented.
Test coverage	**Medium**	Test coverage for Solidity contracts reported by `forge coverage` is `69.25%`. Test coverage for `pallets`, `primitives`, and `runtime-common` reported by `cargo tarpaulin` is `75.20%`. Test coverage for `asset-hubs` and `bridge-hubs` reported by `cargo tarpaulin` is `63.50%`.

Summary of Findings

No	Description	Severity	Status
1	Missing validation for the fee multiplier	Minor	Acknowledged
2	Missing validation for foreignTokenDecimals could lead to a division by zero error	Minor	Acknowledged
3	Missing validation of initialization parameters	Informational	Acknowledged
4	Unused variable in the constructor of the GatewayV2 contract	Informational	Acknowledged

Detailed Findings

1. Missing validation for the fee multiplier

Severity: Minor

The `setPricingParameters` function, defined in `contracts/src/Gateway.sol:375-382`, allows the Gateway contract to update its pricing parameters received from the `system` pallet.

However, since no validation exists for the `multiplier` in both the solidity and the substrate sides of the channel, values in the `(0,1)` range would be accepted and stored in the contract.

Consequently, while the `multiplier` is designed to increase the fees, a value in the `(0,1)` range causes fees to be decreased.

Recommendation

We recommend ensuring that the `multiplier` value is greater than or equal to 1 before storing it in the contract.

Status: Acknowledged

The client states that they are very likely to introduce a new fee system in the next iterations, which will render this parameter obsolete.

They also mention that there are no plans to issue any governance updates that reference this parameter in the meantime.

2. Missing validation for `foreignTokenDecimals` could lead to a division by zero error

Severity: Minor

During the execution of the `constructor` of the Gateway contract, in `contracts/src/Gateway.sol:123`, the `foreignTokenDecimals` input is stored in the contact.

However, since no validation is in place to ensure that `foreignTokenDecimals` is greater than 0, a division by zero error could be thrown in line `460`.

Recommendation

We recommend ensuring that `foreignTokenDecimals` is a non-zero value.

Status: Acknowledged

The client states that they are very likely to introduce a new fee system in the next iterations, which will render this parameter obsolete.

They also mention that there are no plans to issue any governance updates that reference this parameter in the meantime.

3. Missing validation of initialization parameters

Severity: Informational

In `contracts/src/Upgrade.sol:32`, the `Upgrade` contract performs a delegate call including the `initializerParams` variable among its arguments.

However, since the value of the variable is not validated to be non-empty, the call can subsequently fail.

Although there is no direct impact of such a failure, failing fast is best practice for improved code clarity and higher efficiency.

Recommendation

We recommend validating the variable `initializerParams` to not be empty before using it as was implemented in previous versions of the codebase.

Status: Acknowledged

The client states that it is up to the new implementation contract to ensure that it has an initializer function capable of safely accepting any arbitrary initializer parameters.

They emphasize that each implementation contract needs to be carefully developed and, of course, audited to ensure its security.

4. Unused variable in the constructor of the GatewayV2 contract

Severity: Informational

The `constructor` of the GatewayV2 contract, defined in `contracts/src/upgrades/rococo/GatewayV2.sol:10`, takes parameters and then performs a call to the `Gateway` contract.

However, the parameters `recoveryOperator` is not used anywhere in the code and is also not passed to the `Gateway` function call.

Recommendation

We recommend removing or using the `recoveryOperator` parameter.

Status: Acknowledged

The client states that this contract is intended solely for use on their `Rococo` to `Sepolia` testnet and does not impact their production network.