Polygon zkEVM Security Review

March 2023 Engagement

Auditors

Andrei Maiboroda, Lead Security Researcher

Christian Reitwiessner, Lead Security Researcher

Report Version 1

April 6, 2023

Contents

1 About Spearbit

Spearbit is a decentralized network of expert security engineers offering reviews and other security related services to Web3 projects with the goal of creating a stronger ecosystem. Our network has experience on every part of the blockchain technology stack, including but not limited to protocol design, smart contracts and the Solidity compiler. Spearbit brings in untapped security talent by enabling expert freelance auditors seeking flexibility to work on interesting projects together. Learn more about us at spearbit.com

2 Introduction

Polygon zkEVM is a new zk-rollup that provides Ethereum Virtual Machine (EVM) equivalence (opcode-level compatibility) for a transparent user experience and existing Ethereum ecosystem and tooling compatibility.

It consists of a decentralized Ethereum Layer 2 scalability solution utilising cryptographic zero-knowledge technology to provide validation and fast finality of off-chain transaction computations.

This approach required the recreation of all EVM opcodes for transparent deployment and transactions with existing Ethereum smart contracts. For this purpose a new set of tools and technologies were created and engineered, and are contained in this organization.

Disclaimer: This security review does not guarantee against a hack. It is a snapshot in time of Polygon zkEVM according to the specific commit. Any modifications to the code will require a new security review.

3 Risk classification

Severity level	**Impact: High**	**Impact: Medium**	**Impact: Low**
Likelihood: high	Critical	High	Medium
Likelihood: medium	High	Medium	Low
Likelihood: low	Medium	Low	Low

3.1 Impact

- High - leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
- Medium - global losses <10% or losses to only a subset of users, but still unacceptable.
- Low - losses will be annoying but bearable--applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.

3.2 Likelihood

- High - almost certain to happen, easy to perform, or not easy but highly incentivized
- Medium - only conditionally possible or incentivized, but still relatively likely
- Low - requires stars to align, or little-to-no incentive

3.3 Action required for severity levels

- Critical - Must fix as soon as possible (if already deployed)
- High - Must fix (before deployment if not already deployed)
- Medium - Should fix
- Low - Could fix

4 Executive Summary

Over the course of 5 days in total, Polygon Hermez team engaged with Spearbit to review the zkEVM protocol. In this period of time a total of **2** issues were found.

This was a short 5 day engagement with two of the security researchers that focused on reviewing changes in the zkEVM that fixed issues found in the previous two engagements. The overall changes are outlined in the hackmd. The security researchers, on a day-to-day basis, collaborated with the Polygon-Hermez team to review particular changes from the above list.

Summary

Project Name	Polygon
Repository	See above
Type of Project	Zero Knowledge EVM, zk-rollup
Audit Timeline	Mar 6 to Mar 10

Issues Found

Severity	Count	Fixed	Acknowledged
Critical Risk	1	1	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	0	0	0
Gas Optimizations	0	0	0
Informational	1	0	1
Total	**2**	**1**	**1**

5 Findings

5.1 Critical Risk

5.1.1 Incorrect check for calldata overrun in CALLDATACOPY

Severity: Critical Risk

Context: calldata-returndata-code.zkasm#L120

Description: The intention of the code is to check **`offset + size < txCalldataLen`**:

```
; if offset + size is lower than calldata size => length
$               :LT,JMPC(opCALLDATACOPYX0, opCALLDATACOPYloop)
```

But at this point **`A`** contains **`txCalldataLen + size`** and **`B`** contains **`offset`**, so it checks **`txCalldataLen + size < offset`**, which doesn't make sense.

Recommendation: Assign **`offset + size`** to **`A`** and **`txCalldataLen`** to **`B`**.

Polygon-Hermez: We were already aware of this issue, it is reported here and fixed in PR #268.

5.2 Informational

5.2.1 PUSH at end of code can read wrong value

Severity: Informational

Context: stack-operations.zkasm#L168

Description: When, during creation, a **`PUSH`** opcode appears at the end of the bytecode whose data area spans beyond the end of the bytecode, the zkEVM will read the truncated value. For example, if the code ends in **`PUSH4 0x11`**, it will push **`0x11`** onto the stack instead of **`0x11000000`**.

This has no direct effect on correctness, since the next instruction will be **`STOP`** and thus the actual value pushed onto the stack is not relevant.

Recommendation: This could be fixed, but at least it should be documented in the source code.