StarkWare StarkEx Diff Review

Security Assessment

August 11, 2025

Prepared for:
Lotem Kahana
StarkWare

Prepared by: **Guillermo Larregay**

Table of Contents

Project Summary

Contact Information

The following project manager was associated with this project:

Kimberly Espinoza, Project Manager
kimberly.espinoza@trailofbits.com

The following engineering director was associated with this project:

Benjamin Samuels, Engineering Director, Blockchain
benjamin.samuels@trailofbits.com

The following consultant was associated with this project:

Guillermo Larregay, Consultant
guillermo.larregay@trailofbits.com

Project Timeline

The significant events and milestones of the project are listed below.

Date	Event
July 24, 2025	Pre-project kickoff call
July 29, 2025	Delivery of report draft
July 29, 2025	Report readout meeting
August 11, 2025	Delivery of final comprehensive report

Executive Summary

Engagement Overview

StarkWare engaged Trail of Bits to review the security of updates to StarkEx. In particular, we conducted a diff review between versions 3.0.0 and 3.2.0 of the `scalable-dex` contracts in the `starkex-contracts` repository.

One consultant conducted the review on July 28, 2025, for a total of one engineer-day of effort. Our testing efforts focused on the inclusion of the Validium data availability mode and changes to the program output offsets. With full access to source code and documentation, we performed static testing of the code, using automated and manual processes.

Observations and Impact

Several changes were introduced to the perpetual contracts (`StarkPerpetual`, `PerpetualState`, and `UpdatePerpetualState`). The most significant upgrade is the introduction of dual data availability modes, allowing the system to operate in either Validium mode (off-chain data availability with less gas consumption) or Rollup mode (on-chain data availability). New program output parsing logic was implemented to validate the data structures based on the availability mode, which breaks compatibility with old program outputs (TOB-STARKEX-1).

Other changes in the codebase include the addition of support for ERC-1155 tokens while maintaining backward compatibility with existing ERC-20 and ERC-721 assets, refactoring of the `Common` library into the `Addresses` and `StarkExTypes` libraries, and minor changes to other files to improve data validation. We did not identify any issues with these changes.

Recommendations

Based on the findings identified during the security review, Trail of Bits recommends that StarkWare take the following steps:

- **Remediate the findings disclosed in this report.** These findings should be addressed as part of a direct remediation or any refactor that may occur when addressing other recommendations.
- **Create tests for new features.** No tests were available in the repository. It is recommended to have a working test suite that includes unit and integration tests for the protocol and individual features.

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	0
Informational	1
Undetermined	0

CATEGORY BREAKDOWN

Category	*Count*
Data Validation	1

Project Goals

The engagement was scoped to provide a security assessment of updates to the StarkWare StarkEx `scalable-dex` contracts. Specifically, we sought to answer the following non-exhaustive list of questions:

- Is the Validium data availability mode implemented correctly in the contracts?
- Are there any breaking changes between the two versions of the contracts?
- Are data structures correctly validated?

Project Targets

The engagement involved reviewing and testing the following targets.

StarkWare StarkEx scalable-dex Contracts

Repository	https://github.com/starkware-libs/starkex-contracts
Version	Diff between `f3b2506` and `210bd5f`
Type	Solidity
Platform	EVM

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:

- Manual review and static analysis of the changes made in the perpetual contracts (`StarkPerpetual`, `PerpetualState`, and `UpdatePerpetualState`) to support Validium data availability mode
- Manual review of the `Addresses` and `StarkExTypes` libraries, split from the `Common` library
- Manual review of changes in the other files in scope

Coverage Limitations

Because of the time-boxed nature of testing work, it is common to encounter coverage limitations. The following list outlines the coverage limitations of the engagement and indicates system elements that may warrant further review:

- The following are the files that were in scope for the audit. We reviewed only these files and considered only the changes made to these files between commits `210bd5f` and `f3b2506`:
    - `scalable-dex/contracts/src/components/Governance.sol`
    - `scalable-dex/contracts/src/components/GovernanceStorage.sol`
    - `scalable-dex/contracts/src/components/MainGovernance.sol`
    - `scalable-dex/contracts/src/components/MainStorage.sol`
    - `scalable-dex/contracts/src/components/Operator.sol`
    - `scalable-dex/contracts/src/components/TokenRegister.sol`
    - `scalable-dex/contracts/src/components/TokenTransfers.sol`
    - `scalable-dex/contracts/src/components/VerifyFactChain.sol`
    - `scalable-dex/contracts/src/interactions/AcceptModifications.sol`
    - `scalable-dex/contracts/src/interactions/Deposits.sol`
    - `scalable-dex/contracts/src/interactions/TokenAssetData.sol`
    - `scalable-dex/contracts/src/interfaces/BlockDirectCall.sol`
    - `scalable-dex/contracts/src/interfaces/Identity.sol`
    - `scalable-dex/contracts/src/interfaces/MGovernance.sol`
    - `scalable-dex/contracts/src/interfaces/MOperator.sol`
    - `scalable-dex/contracts/src/interfaces/MTokenAssetData.sol`
    - `scalable-dex/contracts/src/interfaces/MainDispatcherBase.sol`
    - `scalable-dex/contracts/src/libraries/Addresses.sol`
    - `scalable-dex/contracts/src/libraries/LibConstants.sol`
    - `scalable-dex/contracts/src/libraries/StarkExTypes.sol`

- scalable-dex/contracts/src/perpetual/ProgramOutputOffsets.sol
- scalable-dex/contracts/src/perpetual/StarkPerpetual.sol
- scalable-dex/contracts/src/perpetual/components/UpdatePerpetualState.sol
- scalable-dex/contracts/src/perpetual/toplevel_subcontracts/PerpetualState.sol
- scalable-dex/contracts/src/tokens/ERC20/IERC20.sol

Summary of Findings

The table below summarizes the findings of the review, including details on type and severity.

ID	Title	Type	Severity
1	Change of program output offsets breaks compatibility with old programs	Data Validation	**Informational**

Detailed Findings

1. Change of program output offsets breaks compatibility with old programs	
Severity: **Informational**	Difficulty: **Low**
Type: Data Validation	Finding ID: TOB-STARKEX-1
Target: `scalable-dex/contracts/src/perpetual/ProgramOutputOffsets.sol`	

Description

The structure of data and offsets in the program output was changed. A new field for data availability mode was added in offset 1, shifting the old fields one position in the output structure.

This makes old program outputs incompatible with the new version, as the old output's `N_ASSET_CONFIGS` is now interpreted as `DATA_AVAILABILITY_MODE`, and the minimum size is increased to nine fields.

Recommendations

Short term, ensure that when contracts are upgraded, operators send the correct version of the program outputs to update the state.

Long term, when changing data structures, consider making the changes backward compatible with the current version by adding new data fields to the end of the structure instead of shifting existing fields.

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

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.

B. Code Quality Recommendations

The following recommendations are not associated with specific vulnerabilities. However, they enhance code readability and may prevent the introduction of vulnerabilities in the future.

- **Fix the following typographical errors:**
  - “Assset” in `TokenAssetData.sol` (lines 134 and 137)
  - “concatanation” in `TokenAssetData.sol` (line 137)
  - “Caclulate” in `UpdatePerpetualState.sol` (line 101)
  - “Pre-caclulate” in `StarkPerpetual.sol` (line 10)
  - “`PROG_OUT_DATA_AVAILABILTY_MODE`” in `ProgramOutputOffsets.sol` (line 8)

About Trail of Bits

Founded in 2012 and headquartered in New York, Trail of Bits 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/trailofbits/publications, with links to papers, presentations, public audit reports, and podcast appearances.

In recent years, Trail of Bits 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 to date with our latest news and announcements, please follow @trailofbits on X or LinkedIn, and explore our public repositories at https://github.com/trailofbits. To engage us directly, visit our "Contact" page at https://www.trailofbits.com/contact or email us at info@trailofbits.com.

Trail of Bits, Inc.
228 Park Ave S #80688
New York, NY 10003
https://www.trailofbits.com
info@trailofbits.com

Notices and Remarks

Copyright and Distribution

Test Coverage Disclaimer

Trail of Bits 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.

Trail of Bits 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.