OPTIMUM
BLOCKCHAIN SECURITY

Reflex Security Assessment

BackrunEnabledSwapProxy

November, 2025

Contents

Disclaimer **2**

About Optimum Blockchain Security **3**

Executive Summary **4**

Overview 4

Contracts Assessed 4

Classification of Issues 5

Findings **6**

1. `swapWithbackrun()` does not handle potential incoming “output tokens” . . . 6

2. `swapWithbackrun()`: transfer of tokens to `msg.sender` is discouraged 7

3. `swapWithbackrun()`: Redundant code 8

Security Best Practices Reference **9**

Disclaimer

This report should not be considered as a security guarantee, investment advice, endorsement or disapproval of any specific project or team. The report makes no claim that the code being reviewed is completely free of vulnerabilities, bugs or potential exploits. Additionally, the report does not assess the financial risk of any asset. Therefore, it is not intended for any third party to make any decisions to buy or sell any asset or product based on this report.

It is important to note that ensuring the security of code is an ongoing process that requires multiple measures. Therefore, it is highly recommended that best coding practices, comprehensive testing, internal audits and bug bounty programs be implemented in addition to this report.

It is the responsibility of the project team to ensure that the code being reviewed is functioning as intended, and that the recommendations provided in this report are thoroughly tested before deployment.

About Optimum Blockchain Security

Optimum Blockchain Security is led by an experienced EVM security researcher with a proven history of delivering high-quality audits and security reviews for leading Web3 protocols. The firm collaborates with top-tier researchers to provide precise, in-depth analyses that go beyond surface-level findings. Our approach emphasizes long-term partnerships, ensuring clients receive not only exceptional technical expertise but also continuous support in strengthening their security posture as their protocols evolve.

For more information, visit **optimumsec.xyz**
Follow updates and insights on **Twitter/X @0xOptimum**

Executive Summary

Overview

The security assessment was made by **one** researcher over a period of **0.5 days**.

Project Name	Reflex
URL	
Code	https://github.com/reflex-mev/reflex
Commit Hash	`ddef31a009a6c801518fac2e7d8038c500717347`
Mitigations Commit Hash	`3c1bd9dbdb8b37cf2451c95e3289acf328f3a37a`
Language	Solidity

Contracts Assessed

Contract Name	Path
BackrunEnabledSwapProxy.sol	core/src/integrations/router/BackrunEnabledSwapProxy.sol

Classification of Issues

Severity	Description
Critical	Issues that may directly result in loss of funds, and thus require an urgent fix.
High	Issues that may not be directly exploitable, or with a limited impact, are still required to be fixed.
Medium	Issues that are not necessarily security vulnerabilities, that are required to be fixed unless there is a clear reason not to.
Low	Subjective issues with a negligible impact.
Info	Subjective issues or observations with negligible or no impact.

Findings

1. `swapWithbackrun()` does not handle potential incoming "output tokens"

Severity: Low
Location: BackrunEnabledSwapProxy.sol#L88

Description:
`swapWithbackrun()` allows the caller to call an arbitrary address with arbitrary parameters through the call to `targetRouter.call()`. The `targetRouter` is supposed to be a DEX that performs a swap and transfer the output tokens to a recipient specified in `swapTxCallData`. The issue here is that there might be `targetRouter` contracts that send the output tokens to `msg.sender` instead of a recipient, in that case the tokens will stay in `BackrunEnabledSwapProxy` and can be stolen by anyone later in a different call to `swapWithbackrun()`.

Recommendation:
Consider adding another parameter - `tokenOut` that represents the token received from the swap to `swapWithBackrun()` and make sure to transfer the balance of this token from `BackrunEnabledSwapProxy` to a `recipient` address specified as a parameter.

Resolution:
Fixed in 3c1bd9db by implementing the auditor's recommendation.

2. `swapWithbackrun()`: transfer of tokens to `msg.sender` is discouraged

Severity: Info
Location: BackrunEnabledSwapProxy.sol#L149-L163

Description:
`swapWithbackrun()` transfers any native token and `tokenIn` back to the `msg.sender` instead of a specified `recipient`.

Recommendation:
Consider adding a `recipient` parameter to `swapWithbackrun()` and use it instead.
Resolution:
Fixed in 3c1bd9db by implementing the auditor's recommendation.

3. `swapWithbackrun()`: Redundant code

Severity: Info

Location: BackrunEnabledSwapProxy.sol#L110-L120, BackrunEnabledSwapProxy.sol#L130, BackrunEnabledSwapProxy.sol#L165-L177, BackrunEnabledSwapProxy.sol#L203-L207

Description:
The function consists of redundant code:

1. The checks of token balance and allowance of the sender.
2. Calling `forceApprove` with 0.
3. The sanity checks that no balance is left in the contract.
4. setting `profits` and `profitTokens` to 0.

Recommendation:
Consider removing this redundant code.

Resolution:
Fixed in 3c1bd9db by implementing the auditor's recommendation.

Security Best Practices Reference

This report references security practices and guidelines from the **Optimum Blockchain Security Guide**. The repository provides a comprehensive and continuously updated collection of best practices for securing smart contracts. Readers are encouraged to review the guide for additional context, rationale, and the latest updates on secure development standards.