

Coinbase: SpendRouter

Security Review

Cantina Managed review by:
0xRajeev, Lead Security Researcher
Cccz, Security Researcher

March 18, 2026

Contents

1 Introduction **2**
1.1 About Cantina . . . 2
1.2 Disclaimer . . . 2
1.3 Risk assessment . . . 2
1.3.1 Severity Classification . . . 2

2 Security Review Summary **3**
2.1 Scope . . . 3

3 Findings **4**
3.1 Informational . . . 4
3.1.1 `SpendRouter` missing `spendWithWithdraw()` routing support . . . 4
3.1.2 Missing zero-address check for `executor` allows permissions that can never be executed . . . 4
3.1.3 Missing zero-address check for `spendPermissionManager` in constructor . . . 5

1 Introduction

1.1 About Cantina

Cantina is a security services marketplace that connects top security researchers and solutions with clients. Learn more at cantina.xyz

1.2 Disclaimer

Cantina Managed provides a detailed evaluation of the security posture of the code at a particular moment based on the information available at the time of the review. While Cantina Managed endeavors to identify and disclose all potential security issues, it cannot guarantee that every vulnerability will be detected or that the code will be entirely secure against all possible attacks. The assessment is conducted based on the specific commit and version of the code provided. Any subsequent modifications to the code may introduce new vulnerabilities that were absent during the initial review. Therefore, any changes made to the code require a new security review to ensure that the code remains secure. Please be advised that the Cantina Managed security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.3 Risk assessment

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

1.3.1 Severity Classification

The severity of security issues found during the security review is categorized based on the above table. Critical findings have a high likelihood of being exploited and must be addressed immediately. High findings are almost certain to occur, easy to perform, or not easy but highly incentivized thus must be fixed as soon as possible.

Medium findings are conditionally possible or incentivized but are still relatively likely to occur and should be addressed. Low findings are a rare combination of circumstances to exploit, or offer little to no incentive to exploit but are recommended to be addressed.

Lastly, some findings might represent objective improvements that should be addressed but do not impact the project's overall security (Gas and Informational findings).

2 Security Review Summary

Base is a secure, low-cost, builder-friendly Ethereum L2 built to bring the next billion users onchain.

From Mar 10th to Mar 11th the Cantina team conducted a review of spend-permissions on commit hash 5f443184. The team identified a total of **3** issues:

Issues Found

Severity	Count	Fixed	Acknowledged
Critical Risk	0	0	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	0	0	0
Gas Optimizations	0	0	0
Informational	3	2	1
Total	**3**	**2**	**1**

2.1 Scope

The security review had the following components in scope for spend-permissions on commit hash 5f443184:

```
src/SpendRouter.sol
```

3 Findings

3.1 Informational

3.1.1 `SpendRouter` missing `spendWithWithdraw()` routing support

Severity: Informational

Context: SpendPermissionManager.sol#L437-L441

Description: `SpendPermissionManager` already supports `spendWithWithdraw()`, which allows a spender to atomically top up an account from `MagicSpend` and then spend.

```
function spendWithWithdraw(
    SpendPermission memory spendPermission,
    uint160 value,
    MagicSpend.WithdrawRequest memory withdrawRequest
) external requireSender(spendPermission.spender) {
    // check spend token and withdraw asset are the same
    if (
        !(spendPermission.token == NATIVE_TOKEN && withdrawRequest.asset == address(0))
            && spendPermission.token != withdrawRequest.asset
    ) {
        revert SpendTokenWithdrawAssetMismatch(spendPermission.token,
        ↪  withdrawRequest.asset);
    }

    // check spend value is not less than withdraw request amount
    if (withdrawRequest.amount > value) {
        revert SpendValueWithdrawAmountMismatch(value, withdrawRequest.amount);
    }

    // check withdraw request nonce postfix matches spend permission hash postfix.
    bytes32 permissionHash = getHash(spendPermission);
    if (uint128(withdrawRequest.nonce) != uint128(uint256(permissionHash))) {
        revert InvalidWithdrawRequestNonce(uint128(withdrawRequest.nonce),
        ↪  uint128(uint256(permissionHash)));
    }

    _useSpendPermission(spendPermission, value);
    _execute({
        account: spendPermission.account,
        target: MAGIC_SPEND,
        value: 0,
        data: abi.encodeWithSelector(MagicSpend.withdraw.selector, withdrawRequest)
    });
    _transferFrom(spendPermission.token, spendPermission.account,
    ↪  spendPermission.spender, value);
}
```

However, `SpendRouter` only wraps `spend` / `approveWithSignature` + `spend`, so flows that rely on `MagicSpend` cannot preserve the router's `executor` / `recipient` binding in a single transaction. As a result, integrators must either build a custom wrapper or fall back to a two-step flow, losing atomicity and weakening the intended periphery UX.

Recommendation: Add router-level wrappers such as `spendWithWithdrawAndRoute()` and, if needed, `spendWithWithdrawAndRouteWithSignature()`, while keeping the existing `executor` and `recipient` checks unchanged.

Coinbase: Accepting suggestion. Fix is implemented in PR 81.

Cantina Managed: Fixed as recommended.

3.1.2 Missing zero-address check for `executor` allows permissions that can never be executed

Severity: Informational

Context: SpendRouter.sol#L87

Description: `decodeExtraData()` validates that `extraData` is exactly 64 bytes but does not check that the decoded executor address is non-zero. In `spendAndRoute()` and `spendAndRouteWithSignature()`, the only post-decode validation is on recipient and executor is used directly in the `msg.sender` comparison without a zero-address guard. Since `msg.sender` can never be `address(0)` in the EVM, a permission accidentally encoded with a zero executor can never be executed.

While `encodeExtraData()` already guards against this for users who go through the helper, users who manually craft `extraData` are susceptible to this scenario.

Recommendation: Consider adding an explicit zero-address check on executor after decoding, either inside `decodeExtraData()` or at the call sites alongside the existing recipient check.

Coinbase: Acknowledged, won't fix. Since a permission thus constructed poses no security risk, we'll avoid placing the extra check on the happy path.

Cantina Managed: Acknowledged.

3.1.3 Missing zero-address check for `spendPermissionManager` in constructor

Severity: Informational

Context: SpendRouter.sol#L65

Description: The constructor accepts a `SpendPermissionManager` argument and assigns it directly to the `PERMISSION_MANAGER` immutable without validating that the address is non-zero. Since `PERMISSION_MANAGER` is immutable, a deployment with a zero address permanently bricks the contract, with no recovery path short of redeployment. Given that `SpendRouter` is intended as a singleton, a misconfigured deployment would require redeployment and redistribution of the new address to all dependent systems.

While the deployment script currently deploys `SpendPermissionManager` immediately before `SpendRouter` and passes the live address directly, it makes an accidental EOA or undeployed address unlikely in practice. However, a zero-address or a zero-code guard is a defensive check for any misconfiguration risk.

Recommendation: Consider adding an explicit zero-address or zero-code guard in the constructor before assigning the immutable.

Coinbase: Accepted. The finding doesn't mention this, but checking for 0 code is a superset of checking 0 address (since the 0 address also won't have any code), and we also add a check for 7702 delegations since that bytecode does not guarantee the persistence of code at the address. Fix is in PR 81.

Cantina Managed: Fix verified.