

# Orbs Spot Audit Report

---

By AstraSec Team  
September 23, 2025

# Scope

---

- ❖ <https://github.com/orbs-network/spot.git>
- ❖ Commit ID: 542e1b3

# N1: Improved Logic of OrderValidationLib::validate()

---

```
function validate(Order memory order) internal view {
    // Validate non-zero critical address
    if (order.reactor == address(0)) revert InvalidOrderReactorZero();
    if (order.executor == address(0)) revert InvalidOrderExecutorZero();
    if (order.exchange.adapter == address(0)) revert InvalidOrderAdapterZero();
    if (order.swapper == address(0)) revert InvalidOrderSwapperZero();

    if (order.deadline <= block.timestamp) revert InvalidOrderDeadlineExpired();
    if (order.chained != block.chainid) revert InvalidOrderChainid();

    if (order.reactor != address(this)) revert InvalidOrderReactorMismatch();
    if (order.input.amount == 0) revert InvalidOrderInputAmountZero();
    if (order.input.amount > order.input.maxAmount) revert InvalidOrderInputAmountGtMax();
    if (order.output.amount > order.output.maxAmount) revert InvalidOrderOutputAmountGtMax();
    if (order.slippage >= Constants.MAX_SLIPPAGE) revert InvalidOrderSlippageTooHigh();
    if (order.input.token == order.output.token) revert InvalidOrderTokenMismatch();
    if (order.output.recipient == address(0)) revert InvalidOrderRecipientZero();
    if (order.exchange.share > Constants.BPS) revert InvalidOrderExchangeShareBps();
}
```

Diagram illustrating the improved logic of `OrderValidationLib::validate()`:

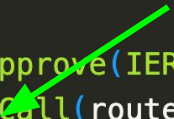
- The initial `if (order.reactor == address(0))` check is marked as **redundant**.
- A green arrow points from the `order.reactor` field in the `if (order.reactor == address(0))` check to the `if (order.reactor != address(this))` check, indicating that the latter check is more comprehensive.
- The `if (order.input.token == order.output.token)` check is highlighted in blue, indicating a new or improved validation step.

## N2: Improved Logic of DefaultDexAdapter::delegateSwap()

---

```
function delegateSwap(bytes32, /*hash*/ uint256, /*resolvedAmountOut*/ CosignedOr  
    external  
    override  
{  
    SafeERC20.forceApprove(IERC20(co.order.input.token), router, co.order.input.a  
    Address.functionCall(router, x.data);  
}
```

Reset approval after swap



## N3: Improved Logic of SurplusLib::distribute()

---

```
/// @dev Distributes surplus tokens between referrer and swapper based on referrer
/// 1. Get total balance of specified token held by this contract
/// 2. Calculate referrer share as (total * shareBps) / BPS
/// 3. Transfer referrer share to ref address (if non-zero)
/// 4. Transfer remaining balance to swapper
/// 5. Emit surplus event if any tokens were distributed
function distribute(address ref, address swapper, address token, uint32 shareBps)
    uint256 total = TokenLib.balanceOf(token);
    uint256 refshare = (total * shareBps) / constants.BPS;
    if (refshare > 0) TokenLib.transfer(token, ref, refshare);
    TokenLib.transfer(token, swapper, total - refshare);
    if (total > 0) emit Surplus(token, swapper, token, total, refshare);
}
```

Annotations in the code:

- A green arrow points from the text "if (total == 0) return;" to the variable `total` in the line `uint256 total = TokenLib.balanceOf(token);`.
- The text "Safely remove" is placed next to the `if (total > 0)` condition in the final line of the function.