CERYVON BUSINESS LOGIC & AUTHORIZATION AUDIT

Synthetic Sample Report

Duplicate Refund Through Cross-Module State Desynchronization

NexaFlow Commerce

This document is a synthetic demonstration only.

It does not represent a vulnerability found in any real company.

Generated: 22 June 2026

Executive Summary

In this synthetic example, the payment records show that the first refund for a delivered order has completed. Because the CRM-side approval-compatible state remains reusable, a second refund request can still reach the final payment operation.

Risk rating	Application	Workflow
High	**NexaFlow Commerce**	**Refund workflow**

This document is a synthetic demonstration only; it does not represent a vulnerability found in any real company.

Scope

The synthetic scope focuses on the refund workflow across the CRM, Order Management, and Payments modules.

- CRM
- Order Management
- Payments
- Refund workflow

Finding

High Risk

Affected Workflow

CRM-approved payment refund for a delivered order

Expected Behavior

The final refund operation should jointly validate CRM approval, order refund eligibility, payment and refund state, prior refund count, tenant boundary, and order ownership.

Observed Behavior

The final operation validates only part of the state and authorization controls required for a safe refund.

Preconditions

- The order belongs to tenant_alpha.
- The order is in the delivered state.
- The Order Management module loads refund eligibility and ownership context.
- A support refund request is opened in CRM and CRM approval is recorded.
- The first refund has completed in the Payments module.
- An approval-compatible state remains available or reusable on the CRM side.

Reproduction Timeline

Step	Actor / Role	Module	State Transition	Result
1	Support	CRM	Refund request is opened	CRM workflow begins
2	System	Order Management	Delivery and ownership context are validated	Order refund eligibility is loaded
3	CRM	CRM	Approval is recorded	Approval-compatible refund state is created
4	System	Payments	First refund is processed	Payment moves to refunded state
5	Support	Payments	Second refund request reaches the final operation	The control set remains incomplete
6	System	Payments	Refund count increases a second time	Duplicate-refund condition is created

Reproduction Steps

1. A support representative opens a refund request in CRM for a delivered order.
2. Order Management loads the delivered state, tenant context, and refund eligibility.
3. Refund approval is recorded in the CRM workflow.
4. Payments processes the first refund and updates the payment state to refunded.
5. A second refund request is prepared while the CRM approval-compatible state remains reusable.
6. The second request reaches the final refund operation and the refund count increases again.

Evidence

Enforced Control

- CRM approval state

Missing or Insufficient Controls

- Prior refund count for the order
- Current payment state
- Whether a refund has already completed
- Order ownership
- Tenant boundary
- Order refund eligibility

In the synthetic model, the final operation enforces one control while six required controls remain missing. After the second final refund step, the refund count reaches 2.

Weak Control Areas

- Prior refund count for the order
- Current payment state
- Whether a refund has already completed
- Order ownership
- Tenant boundary
- Order refund eligibility

Business Impact

- Direct financial loss from a second refund for the same order
- Inconsistent records across CRM, order, and payment systems
- Additional investigation and reconciliation work for operations teams
- Audit-trail integrity and incident-tracing risk

Remediation Recommendations

- Validate CRM, order, and payment state together in one server-side guard at the final refund operation.
- Perform refund creation, payment-state update, and audit logging within an atomic transaction boundary.
- Apply idempotency using the order or payment reference.
- Reject a second refund when a completed refund exists, the payment is already refunded, or the prior refund count is greater than zero.
- Revalidate tenant boundary and order ownership at every final operation.
- Manage critical state transitions through a centralized service.
- Create audit events and alerts for suspicious repeat-refund attempts.

Retest Guidance

- After the first refund completes, attempt a second refund for the same order.
- Verify that the final operation is rejected when CRM approval remains present but the payment state is already refunded.
- Verify that a request from a different tenant context is rejected.
- Confirm that replaying the same order or payment reference does not create a second financial transaction.

Method and Limitations

Ceryvon assessments operate within authorized environments and focus on defined workflows. This example does not claim that every vulnerability would be found or that complete security coverage is provided.

- Applied only within authorized environments and approved scope.
- Focused analysis of defined business workflows.
- Does not guarantee that every vulnerability will be identified.
- This output was created for synthetic demonstration purposes.