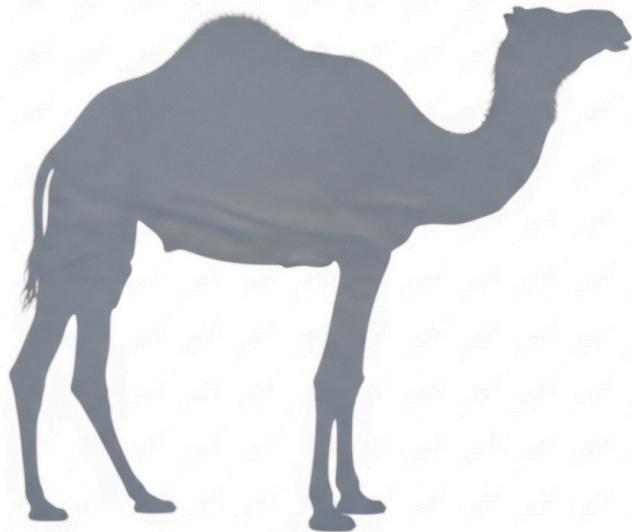


BUG REPORT

Integer Overflow in Fee Calculation



Mundus

02 Dec 2025

Bug Report: QVE-2025-0005

Bug ID	QVE-2025-0005
Finder	<i>Mundus team</i>
Date (reported)	02.12.2025
Status	<i>Found bug</i>
Bug Description	
URL	https://github.com/qubic/core/blob/v1.268.1/src/contracts/Qx.h#L628 https://github.com/qubic/core/blob/v1.268.1/src/contracts/Qx.h#L662 https://github.com/qubic/core/blob/v1.268.1/src/contracts/Qx.h#L791 https://github.com/qubic/core/blob/v1.268.1/src/contracts/Qx.h#L829
Summary	<ul style="list-style-type: none">The multiplication <code>state._price * state._assetOrder.numberOfShares * state._tradeFee</code> can overflow sint64 before the division occurs.
Consequences	<ul style="list-style-type: none">Fee becomes incorrect (wraps to negative or small positive)Seller receives more than they should, major if the fee is negative <code>qpi.transfer(qpi.invocator(), `state._price * state._assetOrder.numberOfShares - state._fee);</code>Contract loses funds, major if the fee is negative <code>qpi.transfer(qpi.invocator(), `state._price * state._assetOrder.numberOfShares - state._fee);</code>Potential for deliberate exploitation with high-value orders
Solution	Use uint128 or restructure calculation
Priority	High
Severity	Critical

Additional Notes, step-by-step Description

Example 1: use existed tokens/SC-shares

- a. Attacker places sell high-value order (i.e. selling 68 QX shares with price 46B qu each)
- b. Attacker places buy order to match with sell order above
- c. Integer overflow occurred, fee becomes negative value
- d. Attacker receives money, QX smart contract loses funds

Example 2: use a new (trash) token

- a. Attacker creates (trash) AAA token (fee 1B qu)
- b. Attacker places sell high-value order of AAA token (no affect to other tokens/SC-shares)
- c. Attacker places buy order to match with sell order above
- d. Integer overflow occurred, fee becomes negative value
- e. Attacker receives money, QX smart contract loses funds