QORTRACE · SMART CONTRACT AUDIT

Test Qorechain

Qorechain

SECURITY	TRUST	FINDINGS
99	60	4
EXCELLENT	MODERATE	0C·0H·0M·1L·3I

TIER	STANDARD
LANGUAGE	SOLIDITY
SUBMISSION	GITHUB
MODEL	claude-sonnet-4-5-20250929
AUDIT ID	7cf6dcc6-ced4-45f0-b862-a25709b91ae8

Executive summary

The source code defines three Solidity interfaces (IQoreAI, IQoreConsensus, IQorePQC) for interacting with QoreChain's on-chain precompiles. These interfaces expose AI risk scoring, anomaly detection, RL consensus parameters, and post-quantum cryptography verification. The code is interface-only (no implementation logic) and appears well-documented with clear NatSpec comments.

Findings by severity

Severity	Open	Resolved	Total
CRITICAL	0	0	0
HIGH	0	0	0
MEDIUM	0	0	0
LOW	1	0	1
INFO	3	0	3
TOTAL	**4**	**0**	**4**

Findings

LOW · 1

LOW Missing events for state-changing observations

`IQorePQC.sol:pqcKeyStatus`

While these are view-only interfaces, the pqcKeyStatus function exposes registration state that may change over time. There is no event mechanism to notify when a PQC key is registered or updated, potentially leading to stale cached data in consumer contracts.

```
function pqcKeyStatus(
    address account
) external view returns (bool registered, uint8 algorithmId, bytes memory pubkey);
```

FIX: If the precompile's underlying state (PQC key registry) can change, consider emitting events from a separate registration contract or document the lack of event support so integrators implement polling or cache-invalidation strategies.

INFO · 3

INFO Interfaces do not enforce precompile address targeting

`IQorePQC.sol:pqcVerify, IQoreAI.sol:aiRiskScore, IQoreAI.sol:aiAnomalyCheck, IQoreConsensus.sol:rlConsensusParams`

The interfaces declare expected precompile addresses (0x0A01, 0x0A02, 0x0B01, 0x0B02, 0x0C01) in NatSpec comments but do not enforce them at the Solidity level. Callers must manually target the correct address when invoking these interfaces.

```
/// @dev Calls the precompile at address 0x0000000000000000000000000000000000000A01.
function pqcVerify(
    bytes calldata pubkey,
    bytes calldata signature,
    bytes calldata message
) external view returns (bool valid);
```

FIX: This is by design for interfaces, but ensure that contracts implementing or calling these functions verify the target address is correct. Consider providing a library wrapper that hardcodes the precompile addresses for safer usage.

INFO No return value bounds documented for risk/anomaly scores

`IQoreAI.sol:aiRiskScore, IQoreAI.sol:aiAnomalyCheck`

The aiRiskScore and aiAnomalyCheck functions return scores in 'basis points' (0–10000), but the interface does not enforce or validate these bounds. Callers must trust the precompile implementation to honor the documented range.

```
/// @return score Risk score in basis points (0 = safe, 10000 = critical risk).
/// @return level Severity level: 0=SAFE, 1=LOW, 2=MEDIUM, 3=HIGH, 4=CRITICAL.
function aiRiskScore(
    bytes calldata txData
) external view returns (uint256 score, uint8 level);
```

FIX: Contracts consuming these interfaces should validate that returned scores are ≤10000 and that level enum values are ≤4 before using them in business logic to prevent unexpected behavior if the precompile malfunctions.

INFO PQC key sizes hardcoded in documentation only

`IQorePQC.sol:pqcVerify`

The pqcVerify function documents expected key and signature sizes (2592 bytes pubkey, 4627 bytes signature for Dilithium-5) in NatSpec, but does not enforce them. Incorrect-length inputs will fail at the precompile level, wasting gas.

```
/// @param pubkey The ML-DSA public key (2592 bytes for Dilithium-5).
/// @param signature The ML-DSA signature (4627 bytes for Dilithium-5).
function pqcVerify(
    bytes calldata pubkey,
    bytes calldata signature,
    bytes calldata message
) external view returns (bool valid);
```

FIX: Consider providing a wrapper function or library that validates input lengths client-side before calling the precompile, or document gas-refund behavior on invalid lengths to guide integrators.

10-point trust check

Check	Result	Evidence
Access Control	PASS	All functions are view-only and do not modify state; no privileged access required.
Events Emitted	FAIL	Interfaces define no events; state-change notifications (e.g., PQC key registration) are not observable on-chain from these interfaces alone.
Reentrancy Protection	PASS	All functions are external view, so reentrancy is not applicable.
Safe Math	PASS	No arithmetic operations present; all return values are computed by precompiles.
Input Validation	FAIL	Interfaces do not validate input bounds (e.g., pubkey/signature lengths, address zero-checks); validation is deferred to precompile implementations.
No Hardcoded Secrets	PASS	No hardcoded secrets, private keys, or sensitive data present in interface definitions.
No Tx Origin For Auth	PASS	No use of tx.origin; all functions are view-only without authentication logic.
Error Handling	FAIL	No revert conditions or error messages defined; precompile failure modes are not documented (e.g., invalid signature, unregistered key).
Documentation Present	PASS	Comprehensive NatSpec comments for all interfaces, functions, parameters, and return values.
Tests Referenced	FAIL	No test files or test references included in the provided source code.