QORTRACE · SMART CONTRACT AUDIT

Qore Chain Test 2 99/100

Qore Chain

SECURITY	TRUST	FINDINGS
98	70	5
EXCELLENT	MODERATE	0C·0H·0M·2L·3I

TIER	STANDARD
LANGUAGE	SOLIDITY
SUBMISSION	GITHUB
MODEL	claude-sonnet-4-5-20250929
AUDIT ID	7ea7494c-a466-4581-98c5-892e7bdf2bd8

Executive summary

Three interface contracts defining precompile interactions for QoreChain's AI risk assessment (0x0B01/0x0B02), RL consensus parameters (0x0C01), and post-quantum cryptography (0x0A01/0x0A02). These are purely declarative interfaces with no executable logic, so typical smart-contract vulnerabilities (reentrancy, access control, state manipulation) do not apply. Security depends entirely on the correctness and determinism of the underlying precompile implementations in the chain runtime.

Findings by severity

Severity	Open	Resolved	Total
CRITICAL	0	0	0
HIGH	0	0	0
MEDIUM	0	0	0
LOW	2	0	2
INFO	3	0	3
TOTAL	**5**	**0**	**5**

Findings

LOW · 2

LOW No validation of precompile return values

`IQoreAI.sol:18-20`

Interfaces declare return types but provide no range or sanity checks. For example, aiRiskScore could return level > 4, or pqcKeyStatus could return algorithmId outside expected values (1 or 2). Consuming contracts must validate these.

```
function aiRiskScore(
    bytes calldata txData
) external view returns (uint256 score, uint8 level);
```

FIX: In consuming contracts, validate that level ≤ 4, score ≤ 10000, and algorithmId ∈ {1,2}. Consider adding helper libraries that wrap these interfaces with built-in validation.

LOW Precompile address collision risk not documented

`IQoreAI.sol:15`

Hardcoded precompile addresses (0x0B01, 0x0B02, 0x0C01, 0x0A01, 0x0A02) are immutable in these interfaces. If the chain upgrades and relocates precompiles, all consuming contracts must be redeployed or upgraded.

```
/// @dev Calls the precompile at address 0x0000000000000000000000000000000000000B01.
```

FIX: Document a governance process for precompile address changes. Consider a registry contract pattern where precompile addresses can be updated via governance, allowing consuming contracts to query current addresses dynamically.

INFO · 3

INFO No executable logic in interface contracts

`IQoreAI.sol:12-26`

These are pure interface definitions with external view functions only. No state variables, no logic branches, and no fund transfers occur in these files. Security analysis must focus on the precompile implementations themselves, which are outside this source.

```
interface IQoreAI {
    function aiRiskScore(
        bytes calldata txData
    ) external view returns (uint256 score, uint8 level);
    function aiAnomalyCheck(
        address sender,
        uint256 amount
    ) external view returns (uint256 anomalyScore, bool flagged);
}
```

FIX: Ensure the underlying precompile implementations at addresses 0x0B01, 0x0B02, 0x0C01, 0x0A01, and 0x0A02 are formally verified and audited in the chain runtime. These interfaces cannot enforce determinism or correctness guarantees.

INFO Precompile gas costs are documented but not enforceable

`IQoreAI.sol:15-16`

Gas cost annotations (e.g., '50,000 flat', '25,000 base + 8 per input byte') are provided in comments but cannot be enforced at the Solidity level. Actual gas consumption is determined by the precompile's EVM implementation.

```
/// @dev Calls the precompile at address 0x0000000000000000000000000000000000000B01.
/// Gas cost: 50,000 flat.
```

FIX: Verify in runtime tests that actual precompile gas costs match documented values. Consider adding a runtime upgrade path if gas costs need adjustment without redeploying consuming contracts.

INFO PQC public key size assumptions in documentation

`IQorePQC.sol:15-16`

Comments specify Dilithium-5 pubkey is 2592 bytes and signature is 4627 bytes, but the interface accepts arbitrary-length bytes. Mismatched sizes could cause verification failure or unexpected behavior in the precompile.

```
/// @param pubkey The ML-DSA public key (2592 bytes for Dilithium-5).
/// @param signature The ML-DSA signature (4627 bytes for Dilithium-5).
```

FIX: In the precompile implementation, enforce exact size requirements and revert with clear error messages if inputs are malformed. Document the error conditions in the interface NatSpec.

10-point trust check

Check	Result	Evidence
Access Control	PASS	Interfaces define only external view functions with no privileged operations; access control is N/A for pure interfaces.
Events Emitted	PASS	No state-changing functions exist; event emission is not applicable to view-only interfaces.
Reentrancy Protection	PASS	All functions are external view (read-only); reentrancy is impossible as no state modification occurs.
Safe Math	PASS	No arithmetic operations are performed in these interfaces; safe math is not applicable.
Input Validation	FAIL	Interfaces declare calldata parameters but provide no bounds checks on return values (e.g., level, algorithmId) or input sizes (e.g., pubkey length).
No Hardcoded Secrets	PASS	No private keys, API keys, or secrets are present in the interface definitions.
No Tx Origin For Auth	PASS	tx.origin is not referenced; all functions are view-only with no authentication logic.
Error Handling	FAIL	Interfaces do not specify revert conditions or error types for precompile failures; consuming contracts must handle failed calls blindly.
Documentation Present	PASS	Comprehensive NatSpec documentation is provided for all interfaces, functions, parameters, and return values, including gas cost estimates.
Tests Referenced	FAIL	No test files or test references are included in the provided source; precompile behavior validation is not demonstrated.