ENS <> OP-Stack Fault Proofs Integration Review

Coinbase Protocol Security

August 18, 2025

coinbase

Contents

Audit Scope

The following contracts were reviewed at commit 128a7c6d36f33b51f3c746eddfc76fdd676b1ae3:

- OPFaultGameFinder.sol
- OPFaultVerifier.sol

Executive Summary

This report covers the scope and findings of the security review performed by the Coinbase Protocol Security team on the ENS <> OP-Stack Fault Proof integration logic. The unruggable-gateways repository was reviewed from Aug 12, 2025 to Aug 13, 2025.

Findings

High Severity

H-01: Dispute games that have not resolved should not be considered usable

In the _isGameUsable function, if the provided dispute game is not blacklisted and is the correct type, the provided minAgeSec is non-zero, and the dispute game is of type Cannon or Permissioned Cannon, then effectively the only additional checks that will be done are:

- Check that the age the dispute game was created is old enough.
- Check if the l2BlockNumber of the dispute game's root claim has been challenged or not and its game status.

If the provided dispute game passes these checks then its corresponding state root will be considered usable for proof validation. However, these checks do not fully guarantee the validity of a state root. A state root should only be considered valid once the dispute game has resolved in favor of the defender, who is the proposer of the state root.

Not waiting for a dispute game to resolve first before using it makes it possible to validate proofs against a future-invalidated state root. This is especially dangerous for chains like Base that have permissionless fault proofs, where anyone at any time can submit a new state root claim.

Medium Severity

M-01: Assumption that gameType will not be greater than 255 is invalid

In _isGameUsable and _gameTypeBitMask, there is an assumption that gameType will never be greater than 255. This is key to the validation logic as it bit-shifts by the value of the gameType. The EVM uses 256-bit words, so a single bit cannot be shifted by a value greater than 255 without overflow occurring.

However, this assumption is invalid as the gameType is a uint32 and there is an existing gameType greater than 255. If the gameType is greater than 255 then _isGameUsable will always return false, but if the Kailua gameType is selected as the respected type then all valid state roots would be rejected.

M-02: Missing check in _isGameUsable

The _isGameUsable function generally follows the validation logic in the OptimismPortal2 contract to check dispute games. These checks include:

- Validating the dispute game is the respectedGameType or is in the provided allowable game types.
- Validating the dispute game has not been blacklisted.
- Validating the dispute game was created after gameTypeUpdatedAt.
- Validating the dispute game resolved in favor of the defender.
- Validating the resolution time of the dispute game has been longer than the finality delay.

However, an additional check has been added to the OptimismPortal2 contract due to code changes around setting a new respectedGameType. Dispute games now determine whether their gameType was the respectedGameType during creation, and the OptimismPortal2 will use this as an additional check to determine whether a dispute game is valid. This check was added because changing the respectedGameType does not require the respectedGameTypeUpdatedAt to be updated at the same time anymore.

Informational

I-01: Outdated comment link

This comment points to the old version of the OptimismPortal contract that does not exist anymore. The comment should point to the new OptimismPortal2 interface and should reference one of the op-contracts release tags instead of pointing to the develop branch.

I-02: Unnecessary game status check when l2BlockNumberChallenged() is true

Checking the game status when l2BlockNumberChallenged() returns true is not necessary, when that boolean is set on a dispute game the challenger will always win.