OP-Challenger explainer

Theop-challenger operates as thehonest actor in the fault dispute system and defends the chain by securing theOptimismPortal and ensuring the game always resolves to the correct state of the chain. For verifying the legitimacy of claims,op-challenger relies on a synced, trusted rollup node as well as a trace provider (e.g.,Cannon).

Specifically,op-challenger performs the following actions:

- monitors and interacts with dispute games
- defends valid output root proposals
- challenges invalid output root proposals
- assistsop-proposer
- by resolving claims and games once chess clocks expire
- claims paid out bonds for both the challenger and proposer

Architecture

This diagram illustrates howop-challenger monitors dispute games, defends valid proposals, and challenges invalid ones. It also shows its interaction with theop-proposer in resolving claims.

Thecannon andop-program executables are run in theop-challenger docker container as sub-processes when required to generate game trace data.

Fault detection responses

op-challenger assesses each claim's validity, countering only those deemed invalid, following this logic:

1. If the trusted node agrees with the output,op-challenger
2. takes no action. An honest challenger does nothing because there are no claims it disagrees with. It continues to monitor the game in case someone posts a counter-claim to the valid root claim, in which case the challenger will participate in the game to defend the proposal.
3. If the trusted node disagrees,op-challenger
4. posts a counter-claim to challenge the proposed output. In contrast to the above, an honest challenger aims to delete any output roots that its trusted node disagrees with in order to claim the bond attached to it. The honest challenger assumes that their rollup node is synced to the canonical state and that the fault proof program is correct, so it is willing to put its money on the line to counter any faults.

Fault dispute game responses

op-challenger iterates through claims as stored in the contract, ensuring ancestors are processed before their descendants. For each claim, the honest challenger determines and tracks the set of honest responses to all claims, regardless of whether that response already exists in the full game state.

Root claim

The root claim is considered to be an honest claim if and only if it has astate witness Hash(opens in a new tab)that agrees with the honest challenger's state witness hash for the root claim.

Counter claims

When a new claim is made in a dispute game, the honest challenger processes it and performs a response. The honest challenger should counter a claim if and only if:

1. The claim is a child of a claim in the set of honest responses
2. The set of honest responses, contains a sibling to the claim with a trace index greater than or equal to the claim's trace index

This implies the honest challenger never counters its own claim, since there is at most one honest counter to each claim, so an honest claim never has an honest sibling.

Possible moves

The challenger monitors each game as new claims are added and reacts according to thehonest actor algorithm(opens in a new tab) to defend valid proposals and invalidate invalid proposals. Amove is a challenge against an existing claim and must include an alternate claim asserting a different trace (e.g., attack, defend, or step).

⚠ See thespecs(opens in a new tab) for the full scope of the honest actor algorithm.

Resolution

When one side of aFaultDisputeGame 's chess clock runs out, the honest challenger's responsibility is to resolve each claim in the game by calling theresolveClaim function on theFaultDisputeGame contract. Once the root claim's subgame is resolved, the challenger then finally calls theresolve function to resolve the entire game.

TheFaultDisputeGame does not put a time cap on resolution - because of the liveness assumption on honest challengers and the bonds attached to the claims they've countered, challengers are economically motivated to resolve the game quickly, thereby liquidating their funds and securing rewards.

Next steps

- Ready to get started? Read our guide on how toconfigureop-challenger on your OP Stack chain
- .
- For more info about howop-challenger
- works under the hood,check out the specs(opens in a new tab)
- .

FAQs

If I don't have a blob archiver with access to historical data, can I lose the game?

Most likely, yes. If nobody has access to the historical data. All the honest actors work together without needing to coordinate offense because they're all trying to play the same moves. So, if there's only one honest actor that lacks a blob archiver, and the actor gets pushed down to the bottom half of the game (~ 32 claims deep in the game), then challenger would log errors and wouldn't be able to respond without the blob archiver. The actor would have 3.5 days on their side of the game to address the problem by switching to a beacon node that does have blobs, and could then proceed as usual.

Note: An actor would only be pushed down to the bottom half of the game if the block being disputed is older than the blob retention period (~18 days). If valid proposals are resolving regularly, this is not possible because each valid proposal becomes the starting point for newly created dispute games. So if there are regular valid proposals, then only ~3.5 days worth of blocks are normally being disputed, which is well within the retention period.

How many CPUs should I run for challenger to work efficiently?

The default--max-concurrency setting suits most operators. Increase it if the challenger lags, or decrease it if it overloads with requests.

How much ETH do you need to challenge in a game?

The honest challengers need to have more combined ETH than the attacker, or they may run out of funds and be unable to respond to games (requiring the security overrides to be used to protect funds). So, there's no strict amount challengers need to have, but here are some general guidelines chain operators can use to estimate:

- Generally speaking, a minimum to play 1 game = bond amount * game max depth + gas costs for each move
- To play one game all the way down to the final step call in a single "chain" costs just over 631.2 ETH in total, so about 315.6 ETH per "side".

What is the bond? What is the bond's role in the FP system?

Eachclaim pays a bond, including the root claim created when the game is created (thus the bond is paid when the game is created). Every time a new claim posts to the game, an additional bond must be paid based on the depth of the claim being posted. It is not necessary to prepay bonds (i.e., its not like staking). Instead, the bond amount is sent as the value of the transaction when callingattack ordefend onFaultDisputeGame or callingcreate onDisputeGameFactory .

Claims that are found to be correct have their bonds refunded. Claims that are found to be incorrect have their bonds paid to the account that posted the left-most uncountered child claim of the incorrect claim. More importantly, the bond for invalid claims is paid to whoever successfully counters the claim, but its setup so that the bond is only ever paid to a single person and never shared.

- There is a delay on claiming bonds of 7 days after the claim is resolved.
- The 7-day period restarts each time a new bond from that game is paid to the same user. Typically this means that bonds are claimable 7 days after the game is resolved.

The calculation for the bond amounts are hard-coded in theFaultDisputeGame contract(opens in a new tab) , and there's agetRequiredBond method on the contract that suggests what bond to use.

How much ETH is required for the challenger bond?

The dispute game factory has the bond value set. To ensure correct game outcomes, the combined funding of all honest actors must be more than the funding available to an attacker. Otherwise the attacker can just post so many claims that the honest actors run out of funds and can no longer counter them. There isn't a fixed amount that guarantees this, so chain operators should have significant funds available at short notice to respond to claims.

Given the guardian can intervene and reallocate bond payments if needed, attackers who try to outspend the honest actors are guaranteed to lose their funds because the guardian will just intervene and pay all their bonds to the honest actors which is a very strong disincentive against trying to win games by outspending people.

FPVM: Cannon MIPS.sol