BREWTFORCE

Sequencer Internal Review

Summary

Project Name	Espresso
Language	Rust
Codebase	https://github.com/EspressoSystems/espresso-sequencer/
Delivery Date	04/10/2024
Team	0xKato, Jarred Parr
Commit(s)	b1f8421cff36f50acd66a5c4df3781f5085ebf7b
Focus points	new_legacy, validate_and_apply_header

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
● Critical	2	0	0	1	0	1
● High	1	0	0	0	0	1
● Medium	1	0	0	0	0	1
● Low	4	3	0	0	0	1

Findings & Resolutions

ID	Title	Category	Severity	Status
HEADER-1	Malicious leader can stop replicas from reading new deposits	Missing Validation	● Critical	Confirmed
REPLAY-1	Replay old builder fee signature	Signature Replay	● Critical	Known issue
CATCH-1	Chain config commitment not validated	Missing Validation	● High	Confirmed
CATCH-2	fetch_accounts return account not validated	Missing Validation	● Medium	Confirmed
Global-1	Inconsistent implementation when handling signatures	Inconsistent Implementation	● Low	Confirmed
Global-2	Lack of comments	Missing Data	● Low	Confirmed
Global-3	Lack of uint tests	Missing Data	● Low	Confirmed
Global-4	Unneeded Panics	Unnecessary Risk	● Low	Confirmed

HEADER-1 | Malicious leader can stop replicas from reading new depositors

Category	Severity	Location	Status
Missing Validation	● Critical	types/src/v0/impls/header.rs	Confirmed

Description

New depositors are kept track of by checking if there are any new deposits from the parent_header by taking a snapshot of the L1 client and checking if there are any new depositors in that time period. This snapshot occurs in both new_marketplace and new_legacy. The problem arises due to the fact that a malicious leader can set the l1_finalized to some arbitrarily large number. This will not cause a problem until that proposal has been voted on, after which the malicious proposal is now the parent.

The malicious proposal now is referred to as the parent, and it will not include any new deposits due to the early return if the new snapshot l1_finalized < parent l1_finalized. The mechanism here is small. Since we are always taking the larger l1_finalized value, we override the proposal's l1_finalized if the new snapshot's l1_finalized < parent l1_finalized so we will always ignore the new depositors in this case.This issue originates in the from_info method in header.rs, there are multiple blocks of if statements here, and the same concept applies to l1.head and timestamp of a proposal.

Recommendation

Check that the proposed L1 finalized block is actually finalized according to their own L1 client, rather than taking it for granted.

Resolution

Fixed in 2089

REPLAY-1 | Replay old builder fee signature

Category	Severity	Location	Status
Replay attack	● Critical	types/src/v0/impls/state.rs:validate_builder_fee	Known issue

Description

Principally located in `types/src/v0/impls/state.rs:validate_builder_fee`, but existing in any location in which `validate_sequencing_fee_signature_marketplace` is called, a replay attack exists such that someone, when calling this method in their sequencer instance, could store the signature that they've seen here and then replay it with the same fee (i.e. call the method twice in the easiest case). This could result in a trivial replay attack against a valid signature that could result in a user being drained over time.

Recommendation

Resolution

Acknowledged in 2018

CATCH-1 | Chain config commitment not validated

Category	Severity	Location	Status
Missing Validation	● High	sequencer/src/catchup.rs:try_fetch_chain_config	Confirmed

Description

If the node missed an upgrade, has the commitment for the chain config that they need, and sends a request to a dishonest peer for the chain config as it doesn't have a full chain config, the peer may respond with any malicious chain config that it pleases. In particular, the principal vector of attack in this case is the `fee_recipient`, which is treated as trusted and can contain any information that is desired.

This is particularly nasty for a couple of reasons:

1. The API endpoint can be innocently changed, meaning that this result can go undetected unless manual intervention takes place.

2. This API endpoint can be changed while maintaining valid functionality in the sequencer node. This sequencer could be fully compliant and participate in the network while simultaneously poisoning all chain configs that its peers request.

It is important to note that this is not the only problem this could also cause the sequencer to fail and potentially block the operation in HotShot that is calling `validate_and_apply_header`, this could be done if the malicious chain config's `max_block_size` is set to 0/lower than the `block_size` or if the `base_fee` is set to 0 any fee will be sufficient to pay.

Recommendation

The issue happens due to the fact that the code assumes that since we request a specific commit than we will receive that specific commit, that is not the case, as there is no validation. To ensure that the returned chain configs commitment matches the commitment we asked for.

Add a check to ensure that the returned commitment matches the requested commitment.

Resolution

Fixed in 2026

CATCH-2 | fetch_accounts return account not validated

Category	Severity	Location	Status
State Violation	● Medium	sequencer/src/catchup.rs:try_fetch_account	Confirmed

Description

In the try_fetch_account method in catchup.rs, the sequencer that is requesting account data from a peer treats this information as trusted. This causes problems when the peer is malicious as it can block the operation being performed needlessly. Consider the use-case in new_legacy.

A malicious actor here could poison the accounts in fetch_accounts by providing an account that is valid but not the requested account which would lead to the requested account being forgotten about.

Recommendation

Check the requested account is indeed the account received.

Resolution

Fixed in [2091]()

Global-1 | Inconsistent implementation when handling signatures

Category	Severity	Location	Status
Inconsistent Implementation	● Low	Global	Confirmed

Description

Within the functions `state.rs` and `header.rs` there exist two differing ways to validate the fees depending on the version of the code, however, in the two places that the validation occurs, we have version checks that differ and, as a result, create an inconsistency with the handling of this logic. The two locations are as follows:

https://github.com/EspressoSystems/espresso-sequencer/blob/main/types/src/v0/impls/header.rs#L423-L4462

https://github.com/EspressoSystems/espresso-sequencer/blob/main/types/src/v0/impls/state.rs#L373-L392

Presumably, it appears that version 1 of this evaluation is the "correct" one, and so the recommended fix is trivial.

Recommendation

Make sure both implementations are performing the same operation. Perhaps via a helper method.

Resolution

Fixed in 2018

Global-2 | Lack of comments

Category	Severity	Location	Status
Missing data	● Low	Global	Confirmed

Description

`validate_and_apply_header` could use a more comprehensive doc comment to ensure that callers within HotShot are not burdened with reading the entire method to understand its behavior. `new_legacy` has minimal comments and no doc comment. As this method is called by HotShot, it would be beneficial to have documentation explaining the use of the method and its potential error conditions.

Recommendation

Add comments that document the code.

Resolution

Global-3 | Lack of unit tests

Category	Severity	Location	Status
Missing data	● Low	Global	Confirmed

Description

Many methods within `validate_and_apply_header` and `new_legacy` lack exhaustive unit tests. For example, `validate_and_apply_header` has no specific tests and, instead, relies in the internal calls to be tested, resulting in its potential errors not being fully covered by different test cases.

Recommendation

Add unit tests.

Resolution

Global-4 | Unneeded Panics

Category	Severity	Location	Status
Unnecessary Risk	● Low	Global	Confirmed

Description

This code panics in two spots in a method which returns a result. This could create an unexpected panic within the HotShot code that calls these methods.

Recommendation

These should propagate the result or None from their respective calls.

Resolution

Disclaimer