Security Audit – Jito MEV Validator

Neodyme AG

September 12, 2022

Contents

Introduction

Jito Labs engaged Neodyme to do a detailed security analysis of their on-chain program. A thorough audit was done between the 4th July and the 9th August 2022.

The audit revealed one medium-severity issue, one low-severity issue and one informational issue.

The following report describes all findings in detail.

Project Overview

Jito Labs brings an MEV engine to Solana, enabling validators to capture MEV for their stakers, while minimizing the negative externalities that MEV brings to the rest of the users and applications running on Solana. For this they modified the Solana validator software, such that MEV searchers can bribe validators to include bundles of transactions in the blocks they produce. This is similar to what Flashbots is doing, but adapted to the architecture of the Solana blockchain.

Scope

We were engaged to audit two smart contracts as well as the relayer and the modified validator software. The block engine is explicitly out-of-scope.

The first smart contract is used by the modified validator to pay out MEV fees to the current leader. The second smart contract uses merkle trees to distribute those fees to the stakers of a given validator. The relayer essentially acts as the validator's TPU port and forwards transactions to the actual modified validator and the block engine over gRPC. The modified validator software has changes to establish a parallel transaction processing pipeline, which processes bundles instead of singular transactions. If one transaction in a bundle fails, all transactions should be rolled back.

Methodology

Neodyme's audit team, which consists of security engineers with extensive experience in Solana smart contract security, reviewed the code of the on-chain contract, paying particular attention to the following:

- Ruling out common classes of Solana contract vulnerabilities, such as:
  - Missing ownership checks,
  - Missing signer checks,
  - Signed invocation of unverified programs,
  - Solana account confusions,
  - Re-initiation with cross-instance confusion,
  - Missing freeze authority checks,
  - Insufficient SPL token account verification,
  - Missing rent exemption assertion,
  - Casting truncation,
  - Arithmetic over- or underflows,
  - Numerical precision errors.
- Checking for unsafe design that might lead to common vulnerabilities being introduced in the future,
- Checking for any other, as-of-yet unknown classes of vulnerabilities arising from the structure of the Solana blockchain,
- Ensuring that the contract logic correctly implements the project specifications,
- Examining the code in detail for contract-specific low-level vulnerabilities,
- Ruling out denial-of-service attacks,
- Ruling out economic attacks,
- Checking for instructions that allow front-running or sandwiching attacks,
- Checking for rug-pull mechanisms or hidden backdoors,
- Checking for replay protection.

Findings

All findings are classified in one of four severity levels:

- **Critical**: Bugs that will likely cause a loss of funds. This means that an attacker can trigger them with little or no preparation or even accidentally. Effects are difficult to undo after they are detected.
- **High**: Bugs which can be used to set up a loss of funds in a more limited capacity, or to render the contract unusable.
- **Medium**: Bugs that do not cause a direct loss of funds but lead to other exploitable mechanisms.
- **Low**: Bugs that do not have a significant immediate impact and could be fixed easily after detection.

Name	Severity	Status
Authentication bypass in relayer	Medium	Resolved
Bundles and transactions from block engine are entirely trusted	Low	Resolved
No fees for failed bundles	Informational	Acknowledged

Authentication bypass in relayer (Medium; Resolved)

Severity	Impact	Affected Component	Status
Medium	DoS	Relayer	Resolved

The relayer opens a gRPC server, to which the modified validator will connect to receive transactions. However, the validator is the only party that is supposed to connect to the relayer, as the gRPC server has no further DoS hardening.

The authentication scheme is as follows: with any request, the validator sends a message and the signature of that message with the validator's private key. This, however, can trivially be spoofed by taking any of the countless signed messages the validator publishes on-chain for voting, bypassing the authentication.

Resolution

The relayer now issues challenges to connecting validators of the form `{validator_pubkey}-{random_challenge}`. The validator then signs this with its private key and sends it back to the relayer, to receive a JWT token. Note that it is infeasible for the challenge creator to create a challenge that forms a valid transaction, as the validator public key overlaps with the recent blockhash field of the transaction.

These changes introduced a potential new DoS vector if the relayer is not configured properly. The challenges are stored server-side, and to prevent DoS attacks through resource exhaustion, the amount of challenges stored at any time is capped to 100,000. Once this limit is reached, no new challenges are issued until challenges expire. Only one challenge is allowed per ip at any time. This works fine for IPv4, as this amount of IPv4 addresses is very expensive. However, this doesn't work anymore if the relayer is configured to bind to IPv6, as 100,000 IPv6 addresses are essentially free.

Bundles and transactions from block engine are entirely trusted (Low; Resolved)

Severity	Impact	Affected Component	Status
Low	Limited loss-of-rewards through validator DoS	Modified Validator	Resolved

The validator receives transactions and bundles from both the relayer and the block engine. The relayer can be entirely trusted by the validator, as it is supposed to be hosted and maintained by the validators themselves. The block engine, however, is not, as it is hosted by Jito Labs or possibly other entities in the future.

To save on computational effort on the validator machine itself, most checks are offloaded to the relayer and the block engine, and the validator does not re-run those checks. These checks include determining whether a transaction is a so-called `simple vote transactions` (a transaction only containing a single vote) and the signature checks. Any block that contains a transaction with a mismatching signature will not be voted on by any other validator, meaning the block will be skipped, costing the validator any potential rewards from that block.

Resolution

The relayer stage inside of the modified validator now has seperate parameters for both transactions coming from the block explorer and the relayer, allowing for both to pass transactions to the sigverify stage instead. This won't verify flags on the packet, but will check the signatures.

Bundles now go through multiple different checks, to make sure that no malformed bundles sent from the block engine can be included in the block:

- All transactions need to deserialize correctly
- Can't have no transactions or too many transactions
- Can't have packets marked for discard
- Can't have invalid signatures
- Can't use consensus accounts or accounts on the blacklist
- Can't have duplicate signatures
- Can't have a transaction with an invalid blockhash
- Can't have a transaction that was recently processed

No fees for failed bundles (Informational; Acknowledged)

Severity	Impact	Affected Component	Status
Informational	Incentive to spam failing bundles	Modified Validator	Acknowledged

When a transaction that is part of a bundle fails, the whole bundle is rolled back. However, the transaction that pays a bribe to the validator is part of the bundle and is therefore rolled back as well. This means failing bundles do not incur a penalty for the submitter, leading to an incentive to spam bundles where the MEV searcher does not know or care whether they will succeed or not. This can lead to similar problems that plagued the Solana blockchain, where the missing incentive to not submit failing transactions lead to so much spam that the blockchain became unusable during high-volatility market environments.

Jito Labs acknowledged this issue and will probably address it at some point in the near future.

Neodyme AG
Dirnismaning 55
Halle 13
85748 Garching
E-Mail: contact@neodyme.io

https://neodyme.io