Bitcoin Core

CVE-2024-52911 - Script Interpreter Remote Crash

After Bitcoin Core 0.14.0 and before Bitcoin Core 29.0, validating a specially-crafted block may cause the node to access previously freed memory.

During validation, necessary data required for checking inputs for each transaction is pre-calculated and cached. For specially crafted invalid blocks, it was possible for this data to be destroyed while it was still being accessed by a background validation thread. An attacker capable of mining a block with sufficient proof-of-work could have exploited this to crash victim nodes. Because of the nature of use-after-free bugs, it is possible that the crash could have been used for remote code execution, though constraints on the input (block) data make this unlikely.

This issue is considered **High** severity.

Details

By default, script validation for new blocks is dispatched to background threads via a vector of `CScriptCheck` functors. Each CScriptCheck holds a pointer to a `PrecomputedTransactionData` object which stores some data needed by each input in the transaction. Because it stores a pointer and not the data itself, care must be taken to ensure that the `PrecomputedTransactionData` outlives the `CScriptCheck`.

The script checks lifetime is enforced by an RAII class, `CCheckQueueControl`. However, the control is intantiated before the precomputed transaction data. Because local objects in C++ are destructed in reverse order of construction, this means the vector of `PrecomputedTransactionData` is destroyed *before* the `CCheckQueueControl`.

This is not an issue when the block is valid, as `CCheckQueueControl::Wait()` will be called before the function returns and the `PrecomputedTransactionData` gets destroyed. However, in case of an early return (when a separate check fails) a background script thread may read the precomputed transaction data after it was destroyed. An attacker could exploit this to crash victim nodes at the expense of a valid PoW at tip.

Attribution

Cory Fields (MIT DCI) discovered this vulnerability and responsibly disclosed it in a detailed report containing a proof of concept for reproduction and a proposed mitigation.

Timeline

- 2024-11-02 Cory Fields privately reports the bug
- 2024-11-06 Pieter Wuille pushes a covert fix to already open PR #31112 which works around the issue by removing the early returns
- 2024-12-03 PR #31112 is merged
- 2025-04-12 Bitcoin Core version 29.0 is released with a fix
- 2026-04-19 The last vulnerable Bitcoin Core version (28.x) goes end of life
- 2026-05-05 Public disclosure.

CVE-2024-52911 - Script Interpreter Remote Crash was published on May 05, 2026.

Bitcoin Core