November 27, 2026 · Changes since V2 (November 18)

LuaVirtualBox V2.1.3

Constant-time call scaling across large scripts, major Med and Max runtime speedups, stronger per-build structural diversity, and compiler reliability fixes.

Since the November 18 V2 update, development focused on eliminating scaling bottlenecks in large protected scripts, cutting per-call memory allocations, hardening output structure across builds, and fixing compiler edge cases for Lua 5.1, Lua 5.4, and Roblox Luau.

Constant-time call scaling and memory reclamation

In earlier V2 builds, entering a protected callback incurred setup cost proportional to the total size of the program, making short, frequent callbacks on large scripts noticeably slower and heavier on memory. V2.1.3 overhauls execution frame reuse and runtime state lifetime management across **Min**, **Med**, and **Max** so call cost stays nearly flat even as scripts grow to thousands of blocks, while releasing state cleanly when closures exit or are collected.

Call latency and allocation by program size (Luau 0.748)

Program size	Before (V2)	V2.1.3	Memory per call (Before → V2.1.3)
60 blocks	10.0 µs	2.3 µs	7.8 KB → 1.06 KB
360 blocks	53.3 µs	2.4 µs	32 KB → 0.99 KB
3,060 blocks	1,181 µs	2.6 µs	392 KB → 0.99 KB
15,060 blocks	5,703 µs	2.6 µs	— → 0.99 KB

Measured on stock Luau 0.748 timing a short callback inside programs of increasing block count. On a 19,871-line Roblox script (10,044 compiled blocks), protected callback cost drops from an estimated ~3.8 ms under the V2 baseline to ~2.6 µs.

Med and Max execution speedups

Extending shared execution frame improvements to **Med** and **Max** dramatically reduced runtime overhead without weakening any protection layer. Across 4,000 rounds of short host, method, and global calls on Windows (median of nine interleaved runs), we measured the following per-round improvements:

Preset	Runtime	Before (V2)	V2.1.3	Speedup
Min	Lua 5.1	16.3 µs	16.8 µs	0.97×
Min	Luau 0.748	7.5 µs	8.2 µs	0.91×
Med	Lua 5.1	346 µs	59 µs	5.86× faster
Med	Luau 0.748	216 µs	23 µs	9.39× faster
Max	Lua 5.1	1,644 µs	1,237 µs	1.33× faster
Max	Luau 0.748	863 µs	553 µs	1.56× faster

Med runs roughly 5.9× faster on Lua 5.1 and 9.4× faster on Luau 0.748. **Min** stays near parity while gaining expanded identifier protection. Actual speed depends on workload, preset, and host environment.

Stronger per-build structural diversity

V2.1.3 expands build-to-build variation across the entire generated output so that independent compilations no longer share predictable structural anchors or uniform helper layouts.

- **Full structure randomization:** Helper layouts, internal data formats, and VM representations vary independently across builds while preserving deterministic reproduction for fixed seeds.
- **Expanded name and macro protection:** Additional name protection now applies across presets, and inline macros such as `LVBOX_NO_VIRTUALIZE`, `LVBOX_ENCSTR`, and `LVBOX_NUMENC` handle captured locals, call arguments, and return lists cleanly.
- **Tighter runtime integrity:** Environment, tampering, and inspection checks were reinforced across presets without regressing Lua 5.1 or Luau compatibility.

Compact emission, compiler fixes, and Emojify

Alongside performance and hardening work, V2.1.3 improves compilation speed, output density, and language correctness:

- **Compact output & faster builds:** Default builds skip unnecessary debug map generation and emit cleaner compact Lua, saving 175 to 686 bytes per build across measured Mini, Min, Med, and Max fixtures while reducing compilation overhead.
- **Compiler correctness fixes:** Resolved wide multi-return and multi-argument register pressure issues on Lua 5.1, corrected statement evaluation ordering in mixed-expression assignments, fixed `2^53` integer precision handling on double-only hosts, and resolved Source Optimizer edge cases.
- **Emojify UI toggle:** Click the **Emojify** toggle in the Web IDE on the site to space Firefly emoji comments and build-time easter eggs safely at token boundaries without altering program execution.

Try V2.1.3 in the Web IDE

LuaVirtualBox V2.1.3 is live on the FireflyProtector Web IDE with the **Min**, **Med**, and **Max** presets, plus the **Source code optimizer**, **Prefer speed over protection** (Max), and **Emojify** toggles in the UI. Because LuaVirtualBox focuses on deep multi-layered VM protection, it remains our strongest and most modern Lua/Luau obfuscator.

Open the LuaVirtualBox V2.1.3 Web IDE · Product overview
