Engine work, with evidence

Ritwik Sharma (NinXyami) · State of Undead Purge, Project Zomboid Build 42 multiplayer · log excerpts are copied unedited, except for player names

---

1. A profiler for every event handler, and the 83-second freeze it caught

What I built. `000_ZombitaProfiler.lua` loads first and wraps every Lua event handler of every mod on the server, timing each by file and line. Each minute it logs the slowest handlers, and it logs straight away when a long stall ends.

What it caught. On 17 Oct 2026, with 22 players online, the server stopped ticking for 83 seconds. My freeze watchdog took a thread dump during the freeze, and the profiler named the exact handler the moment the game came back:

```
14:34:32 FROZEN online file 37 s old (last write 14:33:54 UTC), 22 online, f=81143 +0 in 30 s
14:34:32 dump ~/freeze_dumps/20261017-143431.txt (main: RUNNABLE at StringLib.lower(StringLib.java:717))
14:35:31 freeze over (ticking again) after 83 s

[17-10-26 14:35:17.311] LOG  : Lua  f:81145 st:1,670,542,295> [ZProf] 107s | 102143 ms in handlers | players 22 |
  1) 100085 ms  OnClientCommand  B42Pantherv2:LootZoneStore.lua:335  x778 | 2) 193 ms  LoadGridsquare ...
```

Sources: freeze watchdog log, server DebugLog 14:33-14:36 UTC, thread dump 20261017-143431.

Main thread stack from the dump, top down: `String.toLowerCase` ← `StringLib.lower` ← Kahlua `pcall` ← `Event.trigger` ← `LuaEventManager.triggerEvent (OnClientCommand)` ← `GameServer.receiveClientCommand` ← `GameServer.main`.

Cause. An admin opened the Panther loot-balancer's Loot Zones page. Their client then sent `GetLootZones` 75 times in 102 seconds. Each request copied, summarised and sorted the whole catalogue of 4,583 zones, calling `lower()` on both names in every comparison, and that command was exempt from the mod's own rate limit. The same handler had stalled the server for 148.8 s on 14 Oct. I gave the mod's author the logs, the exact line and the cause.

2. A vanilla Build 42 crash traced into the jar: chicken coops on chunk unload

The crash. Players crashed to desktop while walking near a chicken coop:

```
NullPointerException ... IsoAnimal.removeFromUpdateLists() because "animal" is null
    at IsoHutch.removeFromWorld(IsoHutch.java:1095)
```

What the jar showed (42.21, read class by class). `NetworkPlayerAI.parse(AnimalPacket)` does `animalInside.put(oldPos, null)` and only refills the slot when the packet names a new one. So a hen leaving the coop leaves a `null` on the client. When the chunk unloads, `IsoHutch.removeFromWorld` walks that map and crashes.

A first fix that ran on `OnTick` never fired before a crash. The frame order explains why: `GameClient.update` (packets) runs, then `OnTickEvenPaused`, then `ProcessChunkPos` (the unload, where it crashes), and only then `OnTick`. So the check now runs on `OnTickEvenPaused`, the only hook between the packet and the unload:

```
--- Rebuild one coop's animal list without empty slots. Returns how many empty slots were dropped (0 = untouched).
function HF.clean(hutch)
    local map = hutch:getAnimalInside()
    if not map or map:size() == 0 then return 0 end
    local list = ArrayList.new()
    list:addAll(map:values())
    local keep, empty = {}, 0
    for i = 0, list:size() - 1 do
        local a = list:get(i)
        if a == nil then empty = empty + 1 else keep[#keep + 1] = a end
    end
    if empty == 0 then return 0 end
    map:clear()
    for _, a in ipairs(keep) do pcall(function() hutch:addAnimalInside(a, false) end) end
    return empty
end
...
if Events and Events.OnTickEvenPaused then Events.OnTickEvenPaused.Add(HF.onFrame) end
```

From client/ZX_HutchFix.lua (372 lines, shipped in the live mod). Lua can't use the map's keys (Integer keys reach Lua as Double), so it rebuilds from `values()`.

3. One string over 32 KB crashes every client

Symptom. On 11 Oct 2026 the in-game leaderboard stuck on "Waiting for server..." for everyone, and every client logged `NegativeArraySizeException`. Server-to-client commands write each text value with a 2-byte length, and the rankings JSON had just passed 37 KB as the player count grew.

```
-- 1.7.173: the game sends each text value of a server command with a 2-byte length, so one string over ~32 KB
-- arrives broken ("NegativeArraySizeException" on every client, the board stuck on "Waiting for server..."). The
-- board passed 37 KB on 2026-10-11 as players grew. Big texts now go in pieces of PART_CHARS characters (p1..pn +
n);
-- a small one still goes as data. Kahlua strings are Java strings, so sub() never splits a character, and 10,000
-- characters stay under the limit even if every one needs 3 bytes.
local PART_CHARS = 10000
local function lbArgs(text)
    text = tostring(text or "")
    if #text <= PART_CHARS then return { data = text } end
    local args, n, i = {}, 0, 1
    while i <= #text do
        n = n + 1
        args["p" .. n] = text:sub(i, i + PART_CHARS - 1)
        i = i + PART_CHARS
    end
    args.n = n
    return args
end
```

From server/ZombitaLeaderboardServer.lua. The client joins the parts back together.

4. Removing a mod from a live Build 42 save

Normally you can't take a mod off a running B42 server without breaking it. I found out why by reading the jar, then proved it on scratch dedicated servers on this PC (31 Oct 2026):

• **Fatal for every client:** `ScriptsDictionary` (SpriteConfigs, SpriteOverlayConfigs, ContextMenuConfigs) and `StringDictionary`. `parseLoadList` only clears `isLoaded` for scripts loaded in the current boot, so a saved entry with no script behind it stays loaded for ever. The server boots fine, then every client throws "Missing dictionary script on client".
• **Fix:** a stub entity with the same `module.name`, on both sides. One line parses: `entity X { component SpriteConfig { face N { layer { row = false, } } } }`. A stub pointing at a tile that doesn't exist kills the server instead ("World loading could not proceed").
• **Fatal for one character:** a trait from the removed mod causes an NPE in `CharacterTraits.load`. The fix is to register the same trait from `media/registries.lua`.
• **Not fatal:** removed items and entities, dropped fluids, and tiles that turn into invisible objects.

This is how we removed a vaccine mod from the live server without a wipe.

The scratch-server logs from that test weren't kept. I can re-run it and show it live.