We Terminated a Live GPU Machine Twice on Purpose. Both Jobs Were Replaced on Another Supplier.

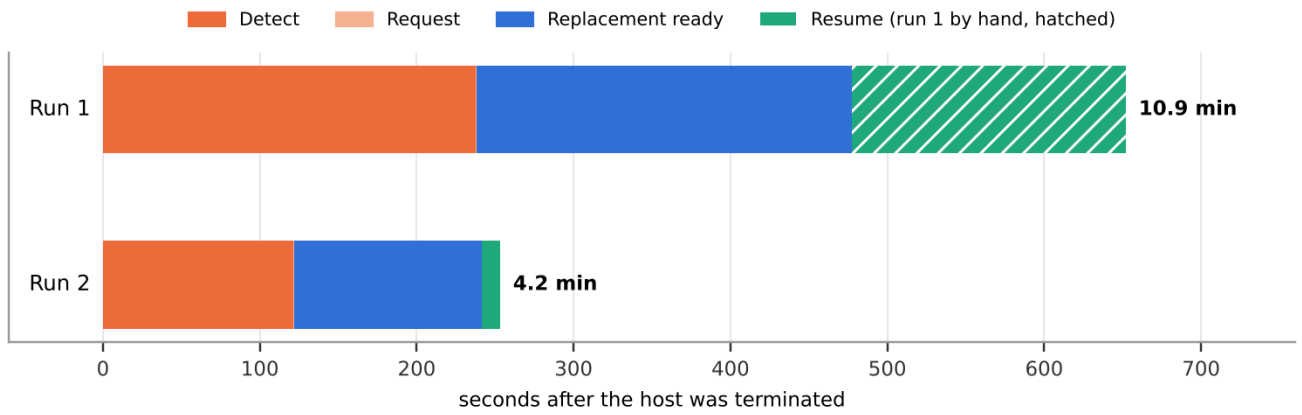
In run 1 the job moved from RunPod to Hyperstack. In run 2 it moved from RunPod to Vast.ai. In run 2 the job resumed from its last saved checkpoint with no one touching it. It took minutes, not milliseconds, and we are publishing the timings.

PREPARED BY KILAWATT CLOUD | OCT. 14, 2026 | DATA: KILAWATT PRODUCTION DATABASE AND ADMIN KILL-TEST EXPORT | INTERNAL TEST ACCOUNT, TWO RUNS, COMPANY-PREPARED, NOT INDEPENDENT REPORTING

2	2 of 2	121 s	4.2 min	seq 4
live machines terminated on purpose	detected and replaced on another supplier	run 2: kill to detect (run 1: 238 s)	run 2: kill to resumed (run 1: 10.9 min, manual)	checkpoint read back intact after the supplier change

On Oct. 14 we ran the failover test we had promised. Using Kilawatt's own test account, we launched a GPU job, let it save checkpoints once a minute, and then terminated the host machine at the supplier while the job was running. Kilawatt noticed the machine was gone, asked a different supplier for a replacement, and the replacement came up healthy. The job's latest checkpoint was still there on the new machine. We ran the test twice, because the first run could not show an automatic resume.

FIGURE 1. Kill to resume, run 1 and run 2

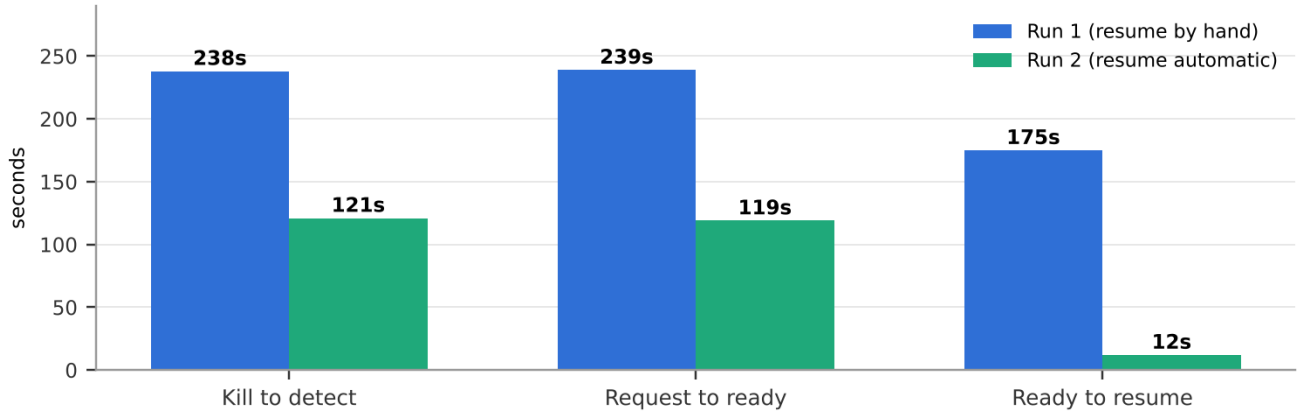


Seconds after the host was terminated. Run 1's resume step was done by hand, because our test script had a bug; run 2 resumed on its own. Source: Kilawatt production database.

THE TWO RUNS SIDE BY SIDE

The Second Run Was Faster at Every Step We Measured

FIGURE 2. Time spent in each phase, in seconds

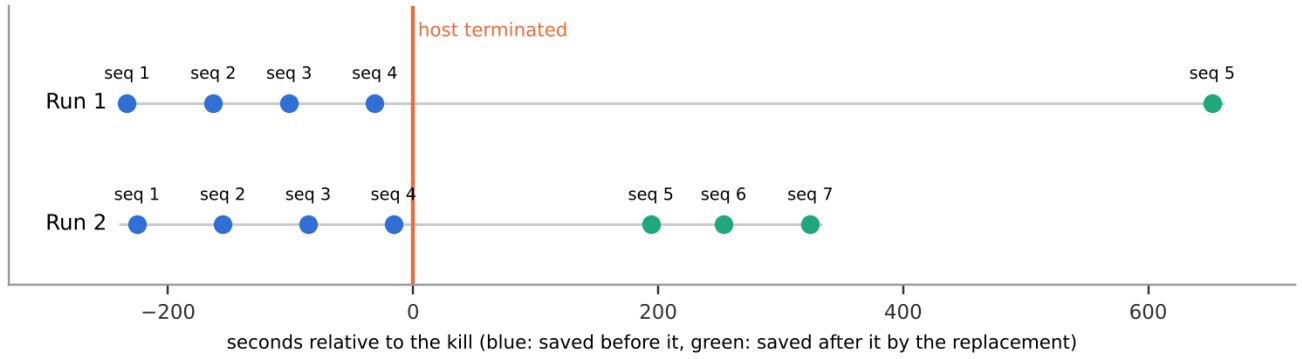


Run 1 did not resume by itself; its 175 seconds is the time until we saved a checkpoint by hand. Run 2's 12 seconds is the server's own stamp. Source: Kilawatt production database.

	Run 1	Run 2
Date, 2026	Oct. 14, 16:50 UTC	Oct. 14, 17:40 UTC
Machine killed on	RunPod	RunPod
Replacement came up on	Hyperstack	Vast.ai
Kill to detect	238 s	121 s
Detect to request	1 s	1 s
Request to replacement ready	239 s	119 s
Ready to resumed	175 s (by hand)	12 s (automatic)
Kill to resumed	10.9 min	4.2 min
Checkpoints before kill / after	4 / 1	4 / 3
Billing paused while down	239 s	119 s

Time-to-ready figures use the server's own timestamps from the admin export. Source: Kilawatt production database.

FIGURE 3. Checkpoints around the kill



Each dot is a checkpoint saved by the job. The last checkpoint before the kill was read back intact from the replacement in both runs. Source: Kilawatt production database.

WHAT IT PROVES

Detect, Replace, Resume From the Last Checkpoint

- **The platform noticed.** Both kills were detected by Kilawatt without anyone telling it. Detection took 238 seconds in run 1 and 121 seconds in run 2.
- **The job moved to a different supplier.** Run 1 went from RunPod to Hyperstack and run 2 from RunPod to Vast.ai.
- **The saved state survived.** Checkpoint 4 was read back from the replacement after both kills, and the replacement went on to save checkpoints 5, 6 and 7 in run 2.
- **Billing paused while the job was down.** The job clock stopped for 239 seconds in run 1 and 119 seconds in run 2.
- **A safety check did its job.** In run 2 the first replacement attempt, at Hyperstack, was declined by our own margin guard. The system moved on and the second attempt, at Vast.ai, succeeded. It is in the attempt log below.

WHAT IT DOES NOT PROVE

The Limits

- **Two runs.** This is not a reliability rate. Two runs on one internal test account, with one simple step-counting job, show the mechanism works. They do not show it works every time.
- **Minutes, not milliseconds.** A replacement machine has to be requested, booted and checked. Expect minutes.
- **Slow detection.** 238 and 121 seconds to notice a dead machine is slower than we want. It is the first thing we will tighten.
- **Two clocks.** In run 2 the test script saw the replacement healthy 176 seconds after the kill, while the server's own ready stamp came about 65 seconds later, at 241 seconds. The report uses the server's stamps.
- **Not independent.** We ran the test on our own account and prepared this report ourselves.

APPENDIX A

The Run Log, Including the Run We Threw Out

Run	Job ID	Path	Result
1	40c7e852	RunPod to Hyperstack	Passed detect, replace, checkpoint survival. Resume timed by hand (script bug, fixed).
2	591a881f	RunPod to Vast.ai	Passed, including automatic resume from checkpoint 4.
Aborted	ab191092	Vast.ai, no kill test	Excluded. Machine took 14.6 minutes to boot, which used most of the 20-minute job, so we stopped it. Not a test result.
Refused	none created	RunPod A100	Supplier reported no stock before launch. No machine, nothing to test. We retried and it succeeded.

Attempt log, as recorded by the platform

Run	#	Supplier	Outcome	Note
1	1-2	RunPod	success	Initial launch (forced to RunPod, test account only)
1	3	Hyperstack	success	Rescue after host drop
2	1-2	RunPod	success	Initial launch (forced to RunPod, test account only)
2	3	Hyperstack	failure	Blocked by margin guard
2	4	Vast.ai	success	Rescue after host drop
Aborted	1-2	Vast.ai	success	Initial launch, later stopped

APPENDIX B

How the Numbers Were Produced

- **Timings.** Taken from the platform's own timestamps for kill, detect, replacement request, replacement ready and resume, shown in the admin kill-test export and checked against the production database.
- **Test script.** A small script saved one checkpoint about every 60 seconds. We terminated the host at the supplier with the admin kill button after checkpoint 4.
- **Bug and fix.** Our first script waited for a state the platform never reports, so it missed the resume in run 1. We fixed it and re-ran.

DISCLOSURES

This report was prepared by Kilawatt Cloud about its own service and is not independent reporting. Both runs used an internal Kilawatt test account. No outside customer is involved. It is two runs, not a reliability rate, and it contains no forecast and no offer. Kilawatt Cloud is a trade name of Right Recruit LLC.