A graph of a bar graph

AI-generated content may be incorrect.

For this test, I chose P95 latency, a performance metric that indicates the latency value below which 95% of requests are completed. The second metric that I used for testing is throughput (RPS) to measure the number of requests a system can handle within a one-second interval, to test a systems ability to handle traffic, reflecting how quickly it can process requests.

**1 Latency (P95)**

* The containerized deployment consistently shows **10 – 30 % higher P95 latency**.
* Absolute overhead ranges from **+2 ms (8 users)** to **+60 ms (128 users)**.
* This is typical of Docker’s extra network bridge and cgroup scheduling.

**2 Throughput (RPS)**

* Differences are **within ±2 %**—statistically indistinguishable without deeper tests.
  + At higher concurrencies (64, 128) the container variant is **slightly faster**, likely because the Linux scheduler evens out CPU bursts when the process is cgroup-confined.

**3 Overall**

* **Performance cost is modest**: small latency penalty, negligible throughput impact.
* Given the gains in portability, reproducibility, and CI/CD automation, containerization looks **justifiable** for this MCP workload.

**Conclusion:** For serving LLMs via a containerized MCP architecture, the results confirm that containerization introduces **slightly higher latency** but **does not degrade generation performance or concurrency handling**. This means I can confidently containerize LLM services—especially when reproducibility, deployment automation, or security isolation are required.