LA-UR-22-27687

# Charliecloud's Git-based Cache is Competitive with Alternatives

Zechariah Noah Hounshel, Ashlyn Lee, Benjamin Stormer

## Background

The essential software stack of an application is often bundled together in a "container" for ease of portability and use. Systems for building containers frequently use a build cache to decrease build time. Two industry standard container build systems are Podman and Docker. Standard Docker requires its users to have administrator privileges in order to build and run containers, and while Podman contains a "rootless" mode, it is not fully unprivileged as it uses setuid executables.

Charliecloud is a LANL-developed container build system designed for unprivileged use. Docker and Podman use a build cache based on the OverlayFS file system, whereas Charliecloud uses a Git-based cache. Charliecloud's Git-based cache is a recent development and its performance is largely untested. We measured the build speed of container images for Charliecloud, Docker, and Podman to determine the practicality of Charliecloud's Git-based build cache.

## Methods

We built six container images of differing size and complexity using Charliecloud, Docker, and Podman. We built the images with varying levels of the build cache being "filled" in the respective systems, including with Charliecloud's cache disabled. Hyperfine, a command line benchmarking tool, was used to run each test repeatedly and gather time data from each run.

## Results

We found that Charliecloud's cache generally resulted in slower build times than Docker and Podman's but was not meaningfully slower than Podman's. We also found that Podman resulted in the most inconsistent build times of the three technologies.

On our compute nodes, we found that Charliecloud was roughly 2 times slower than both Docker and Podman for cold cache tests, with the average Charliecloud build time being 290 seconds (averaged across all tests). For hot cache tests, Charliecloud was roughly 2 times slower than Podman, but 10 times slower than Docker with the average Charliecloud build time being roughly 5 seconds (averaged across all tests).

## Conclusions

Our results suggest that Charliecloud's Git-based cache system is a viable form of container build caching. Charliecloud's cache has room for several optimizations, but was not so much slower than Podman's caching system to be problematic in most workflows. Potential optimizations involve fixing a bug that fails to cache "COPY" commands and checking Dockerfile hashes to reduce Git tree traversal time.