Docker and Ambassador Labs Announce Telepresence for Docker, Improving the Kubernetes Development Experience

This article was fetched from an rss feed(23/03/2023)

I've been a long-time user and avid fan of both Docker and Kubernetes, and have many happy memories of attending the Docker Meetups in London in the early 2010s. I closely watched as Docker revolutionized the developers' container-building toolchain and Kubernetes became the natural target to deploy and run these containers at scale.

Today we're happy to announce Telepresence for Docker, simplifying how teams develop and test on Kubernetes for faster app delivery. Docker and Ambassador Labs both help cloud-native developers to be super-productive, and we're excited about this partnership to accelerate the developer experience on Kubernetes.

What exactly does this mean?

- When building with Kubernetes, you can now use Telepresence alongside the Docker toolchain you know and love.
- You can buy Telepresence directly from Docker, and log in to Ambassador Cloud using your Docker ID and credentials.
- You can get installation and product support from your current Docker support and services team.

Kubernetes development: Flexibility, scale, complexity

Kubernetes revolutionized the platform world, providing operational flexibility and scale for most organizations that have adopted it. But Kubernetes also introduces complexity when configuring local development environments.

We know you like building applications using your own local tools, where the feedback is instant, you can iterate quickly, and the environment you're working in mirrors production. This combination increases velocity and reduces the time to successful deployment. But, you can face slow and painful development and troubleshooting obstacles when trying to integrate and test code into a real-world application running on Kubernetes. You end up having to replicate all of the services locally or remotely to test changes, which requires you to know about Kubernetes and the services built by others. The result, which we've seen at many organizations, is siloed teams, deferred deploying changes, and delayed organizational time to value.

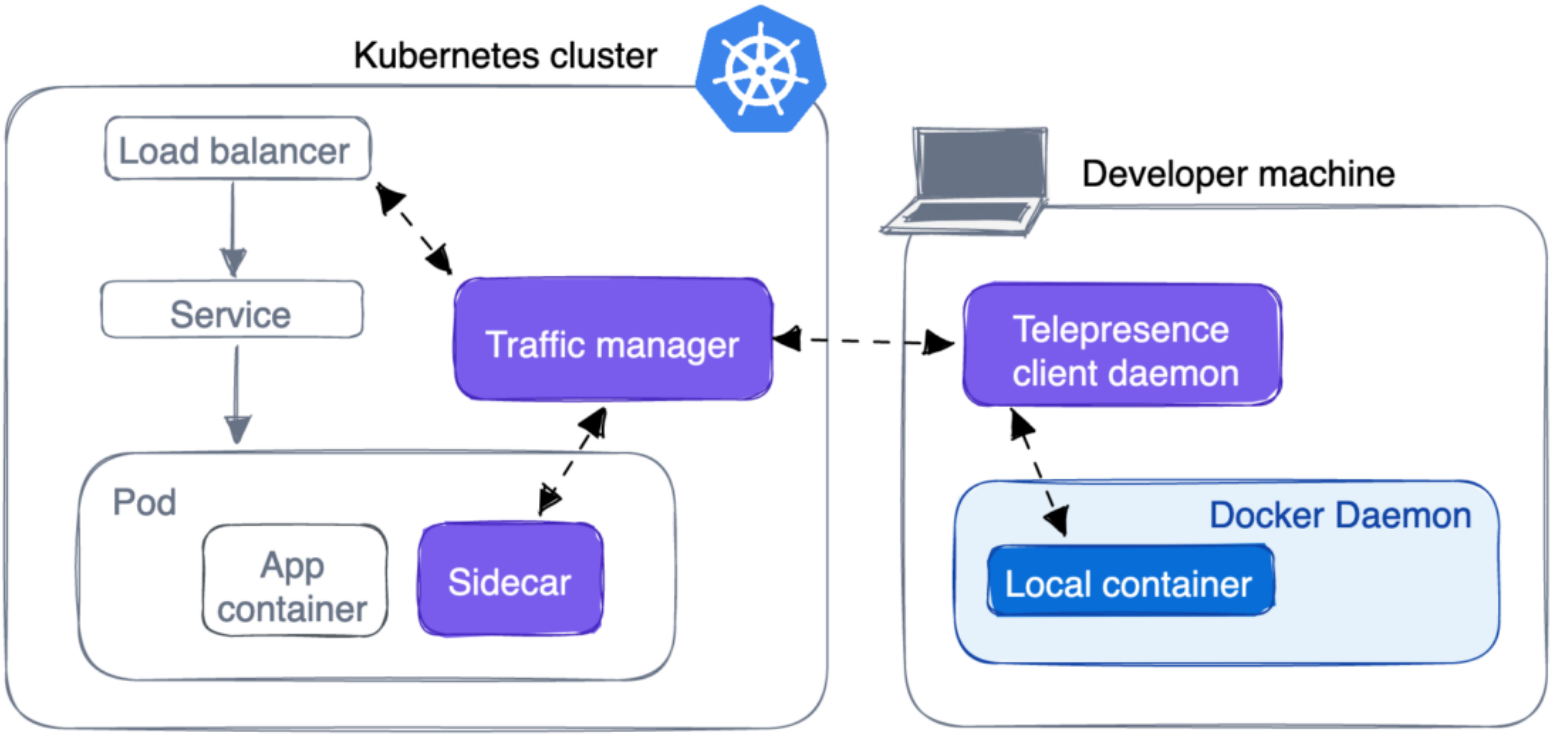
Bridging remote environments with local development toolchains

Telepresence for Docker seamlessly bridges local dev machines to remote dev and staging Kubernetes clusters, so you don't have to manage the complexity of Kubernetes, be a Kubernetes expert, or worry about consuming laptop resources when deploying large services locally.

The remote-to-local approach helps your teams to quickly collaborate and iterate on code locally while testing the effects of those code changes interactively within the full context of your distributed application. This way, you can work locally on services using the tools you know and love while also being connected to a remote Kubernetes cluster.

How does Telepresence for Docker work?

Telepresence for Docker works by running a traffic manager pod in Kubernetes and Telepresence client daemons on developer workstations. As shown in the following diagram, the traffic manager acts as a two-way network proxy that can intercept connections and route traffic between the cluster and containers running on developer machines.



Once you have connected your development machine to a remote Kubernetes cluster, you have several options for how the local containers can integrate with the cluster. These options are based on the concepts of *intercepts*, where Telepresence for Docker can re-route — or intercept — traffic destined to and from a remote service to your local machine. Intercepts enable you to interact with an application in a remote cluster and see the results from the local changes you made on an intercepted service.

Here's how you can use intercepts:

- **No intercepts:** The most basic integration involves no intercepts at all, simply establishing a connection between the container and the cluster. This enables the container to access cluster resources, such as APIs and databases.
- **Global intercepts:** You can set up global intercepts for a service. This means all traffic for a service will be re-routed from Kubernetes to your local container.
- **Personal intercepts:** The more advanced alternative to global intercepts is personal intercepts. Personal intercepts let you define conditions for when a request should be routed to your local container. The conditions could be anything from only routing requests that include a specific HTTP header, to requests targeting a specific route of an API.

Benefits for platform teams: Reduce maintenance and cloud costs

On top of increasing the velocity of individual developers and development teams, Telepresence for Docker also enables platform engineers to maintain a separation of concerns (and provide appropriate guardrails). Platform engineers can define, configure, and manage shared remote clusters that multiple Telepresence for Docker users can interact within during their day-to-day development and testing workflows. Developers can easily intercept or selectively reroute remote traffic to the service on their local machine, and test (and share with stakeholders) how their current changes look and interact with remote dependencies.

Compared to static staging environments, this offers a simple way to connect local code into a shared dev environment and fuels easy, secure collaboration with your team or other stakeholders. Instead of provisioning cloud virtual machines for every developer, this approach offers a more cost-effective way to have a shared cloud development environment.

Get started with Telepresence for Docker today

We're excited that the Docker and Ambassador Labs partnership brings Telepresence for Docker to the 12-million-strong (and growing) community of registered Docker developers. Telepresence for Docker is available now. Keep using the local tools and development workflow you know and love, but with faster feedback, easier collaboration, and reduced cloud environment costs.

You can quickly get started with your Docker ID, or contact us to learn more.