Kubernetes is Deprecating Docker?!

When people think of containers, they think of Docker and Kubernetes. Docker has been the big name when it comes to building and running containers, and Kubernetes has been the big name when it comes to managing and orchestrating them. It might seem a bit shocking to hear that Kubernetes is deprecating support for Docker as a container runtime starting with Kubernetes version 1.20.

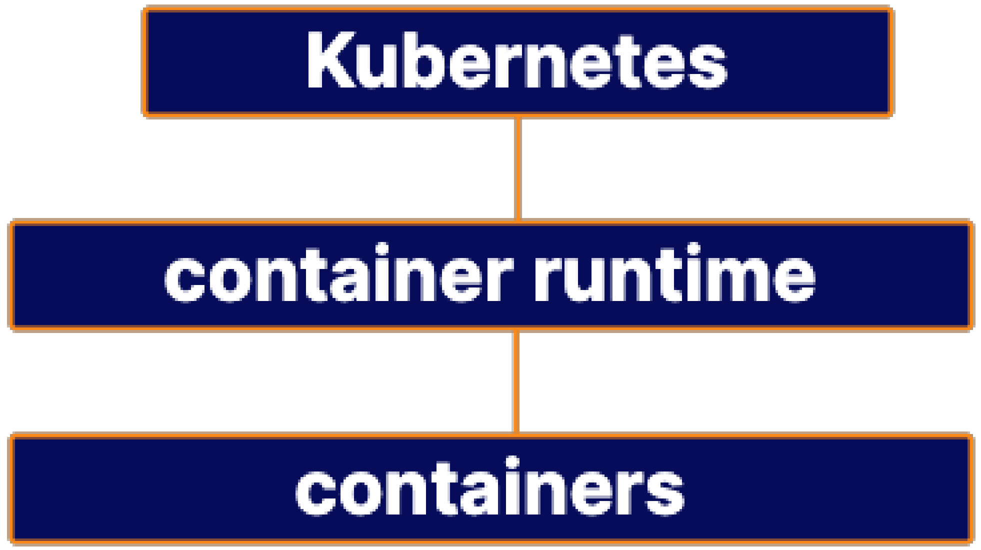
Mithun Technologies
devopstrainingblr@gmail.com
Ph: +91-9980923226

What’s Changing with Docker?

Mithun Technologies
devopstrainingblr@gmail.com
Ph: +91-9980923226

- Kubernetes deprecating Docker is actually not as big of a deal as it sounds.

Kubernetes is removing support for Docker as a **container runtime**. Kubernetes does not actually handle the process of running containers on a machine. Instead, it relies on another piece of software called a **container runtime**.



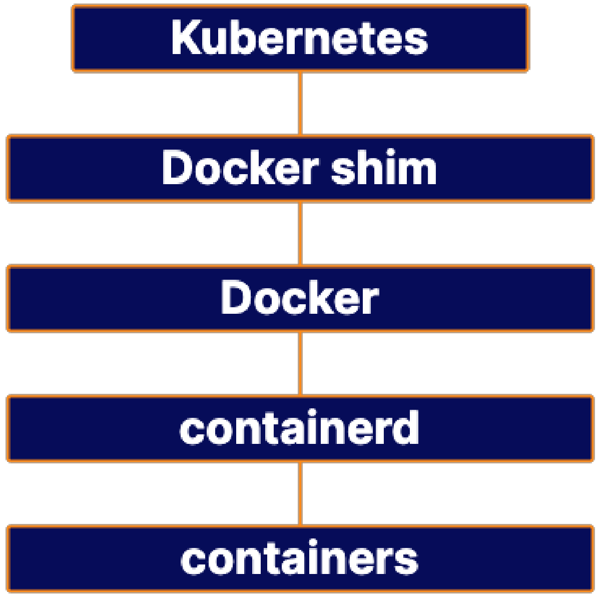
- The container runtime runs containers on a host, and Kubernetes tells the container runtime on each host what to do.
- You can actually choose from a variety of options when it comes to what software you want to use as your container runtime when running Kubernetes.
- Up to now, popular option was to use Docker as the container runtime.
- **However, this(Docker) will no longer be an option as container run time for Kubernetes in the future.**

Why is Kubernetes Deprecating Docker?

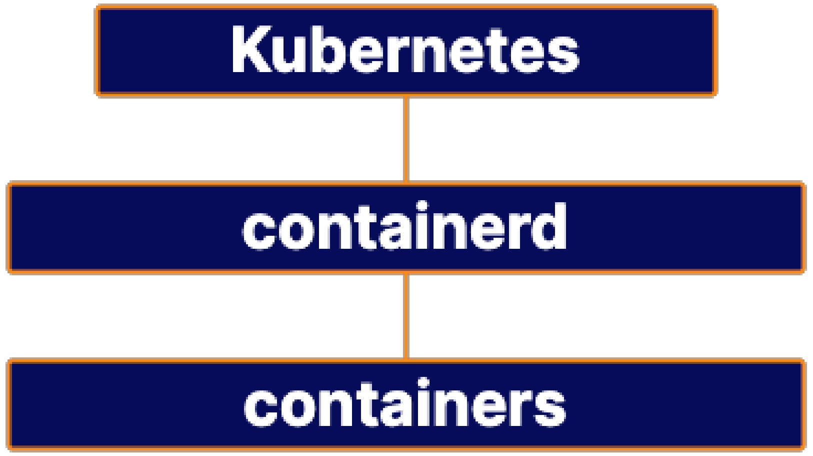
- Kubernetes works with all container runtimes that implement a standard known as the **Container Runtime Interface (CRI)**. This is essentially a standard way of communicating between Kubernetes and the container runtime, and any runtime that supports this standard automatically works with Kubernetes.
- **Docker does not implement the Container Runtime Interface (CRI).** In the past, there weren’t as many good options for container runtimes, and Kubernetes implemented the Docker shim, an additional layer to serve as an interface between Kubernetes and Docker. Now, however, there are plenty of runtimes available that implement the CRI, and it no longer makes sense for Kubernetes to maintain special support for Docker.
- **Docker is not actually a container runtime**! It’s actually a collection of tools that sits on top of a container runtime called **containerd**.

Why is Kubernetes Deprecating Docker?

Docker does not run containers directly. It simply creates a more human-accessible and feature-rich interface on top of a separate, underlying container runtime. When it is used as a container runtime for Kubernetes, Docker is just a middle-man between Kubernetes and containerd.



However, **Kubernetes can use containerd directly as a container runtime, meaning Docker is no longer needed in this middle-man role**. Docker still has a lot to offer, even in a Kubernetes ecosystem. It's just not needed specifically as a container runtime.



What’s the Role of Docker Going Forward?

- Although Docker is not needed as a container runtime in Kubernetes, it still has a role to play in the Kubernetes ecosystem, and in your workflow.
- Docker is still going strong as a tool for developing and building container images, as well as running them locally.
- Kubernetes can still run containers built using Docker’s **Open Container Initiative (OCI)** image format, meaning **you can still use Dockerfiles and build your container images using Docker**.
- **Kubernetes will also continue to be able to pull from Docker registries** (such as Docker hub). This means that Docker will remain a powerful contender when it comes to managing the images once they are built.
- Docker will continue to be a useful tool on your development workflows and continuous integration (CI) systems, even if you don’t need it to run your containers underneath in Kubernetes.

Docker Alternatives

- If you are currently using Docker as a container runtime in your Kubernetes environment, you will need to make some changes. Moving forward, you can simply eliminate Docker as a middle-man in your Kubernetes environment. Instead, use another container runtime, such as containerd or CRI-O.
- Before upgrading to a Kubernetes version removes support for Docker (which is currently estimated to release in late 2021), you will need to modify (or replace) existing Kubernetes nodes so that they use a supported container runtime other than Docker. Starting now, you may want to start building any new nodes so that they use a non-Docker container runtime as well.
- Other than that, nothing is really changing. You can continue to use Docker to build your images, as well as to run containers locally for development purposes, or in your continuous integration (CI) stack. You can also continue to use Docker registries to store and manage your images.

Questions ?