KUBERNETES ARCHITECTURE

Kubernetes Architecture

Pods

The whole point of Kubernetes is to orchestrate the lifecycle of a container. We do not interact with particular containers. Instead, the smallest unit we can work with is a *Pod*. Some would say a pod of whales or peas-in-a-pod. Due to shared resources, the design of a Pod typically follows a one-process-per-container architecture.

Containers in a Pod are started in parallel. As a result, there is no way to determine which container becomes available first inside a pod. The use of **`InitContainers`** can order startup, to some extent. To support a single process running in a container, you may need logging, a proxy, or special adapter. These tasks are often handled by other containers in the same pod.

There is only one IP address per Pod, for almost every network plugin. If there is more than one container in a pod, they must share the IP. To communicate with each other, they can either use IPC, the loopback interface, or a shared filesystem.

While Pods are often deployed with one application container in each, a common reason to have multiple containers in a Pod is for logging. You may find the term **`sidecar`** for a container dedicated to performing a helper task, like handling logs and responding to requests, as the primary application container may not have this ability. The term **`sidecar`**, like **`ambassador`** and **`adapter`**, does not have a special setting, but refers to the concept of what secondary pods are included to do.