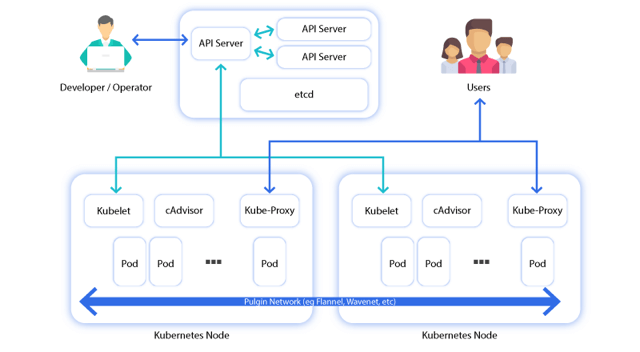
**What is Kubernetes**

* An open-source tool for container orchestration
* Automates the process of deploying and managing multi-container applications at scale
* A portable, extensible platform for managing containerized workloads and services that facilitates both declarative configuration and automation
* K8s provides a container-centric management environment
* Google open-sourced Kubernetes project in June 2014.
* K8s can be thought of as:

1. A container platform that manages clusters of containers
2. A Micro-services platform
3. A portable cloud platform and lot more

* K8s is organized around Master node and Worker nodes
  + Kubernetes Master is the primary control unit that manages workload and communication across the system. Its components are:
    - Etcd storage (key-value data store for config data, accessed by all nodes of the cluster)
    - API-server (central management entity that receives REST requests for modifications)
    - Scheduler (schedule the pods on various nodes based on resource utilization)
    - Controller manager (runs a number of distinct controller process in the background)



* + Worker nod, also known as minion node, contains information to manage networking between containers. Its components are
    - Kubelet (ensures all containers in the node are running and in a healthy state)
    - Container (lowest level of microservice, placed inside pods)
    - Kube Proxy (acts as a network proxy and load balancer)
    - cAdvisor (responsible for monitoring and gathering data about resource usage on a node)

**Master and slave**

* Master is the brain of k8s cluster. Here we have a core API server that maintains RESTful web services for querying and defining our desired cluster and workload state.
* Master includes a scheduler that works with the API server
* Public cloud (Anyone can use it)

**Container concepts**

At the core of container technology are

* Control groups (cgroups)
* Namespaces
* Additionally, Docker uses union filesystems for added benefits to the container development process.

**Docker:** A tool to create an image (Dockerimage) of an application and the dependencies needed to run it by using a configuration file (Dockerfile). A dockerfile has all the instructions on how to build the final image for deployment and distribution. The image is then used by k8s for deployment.

**Cgroup:** Restricts resources a process can consume (e.g. CPU, Memory, Disk IO etc.). It works by allowing the host to share and also limit the resources each process or container can consume. This allows several containers to share CPU and memory while staying within the predefined constraints.

**Namespace:** Change a process’s view of the system (e.g. network interfaces, PIDs, users, mount etc.) It offers another form of isolation for process interaction within operating systems. Container processes are limited to see only what is in the same namespace.

**Union filesystems:** Are also a key advantage of using Docker containers. Containers run from an image. Much like an image in the VM or cloud world, it represents state at a particular point of time. Container images snapshot the filesystem but tend to be much smaller than a VM.

Deployment of containers are based on operating-system-level virtualization rather than hardware virtualization. These containers are isolated from each other and from the host. They have their own filesystems, they cannot see each other’s processes and their computational resource usage can be bounded. They are portable across clouds and OS distributions.

**Points to remember**

* Continuous integration is a development practice that requires developers to integrate code to a shared repository several times a day. The result is that updates and bug fixes happen much faster and the overall quality improves
* Using Docker ensures that OS, packages and application versions are the same across development, staging and production environments
* Micro-service is an architectural style – an approach to develop a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms. These services are built around business capabilities and independently deployable by fully automated deployment machinery.
* Kubernetes (k8s) is a management and orchestration tool for containerized application deployment.
* K8s operates at container level rather than at hardware level. If an application can run in a container, it should run great on k8s
* K8s control script is known as ‘kubectl’. Most administrative interactions are done via this script
* The ideas of actual state and desired state is the key of how k8s manages the cluster and its workloads. All the pieces of k8s are constantly working to monitor the current actual state and synchronize it with the desired state defined by the administrators via the API server or kubectl script. There will be times when these states do not match up, but the system is always working to reconcile the two
* By default, Kubernetes uses the ‘gce’ provider for Google cloud. We can override this default by setting the KUBERNETES\_PROVIDER environment variable. Some permissible values are listed below:
  + gce (Google Compute Engine – public cloud)
  + gke (Google Conatiner Engine – public cloud)
  + aws (Amazon Web Services – public cloud)
  + azure (Microsoft Azure – public cloud)