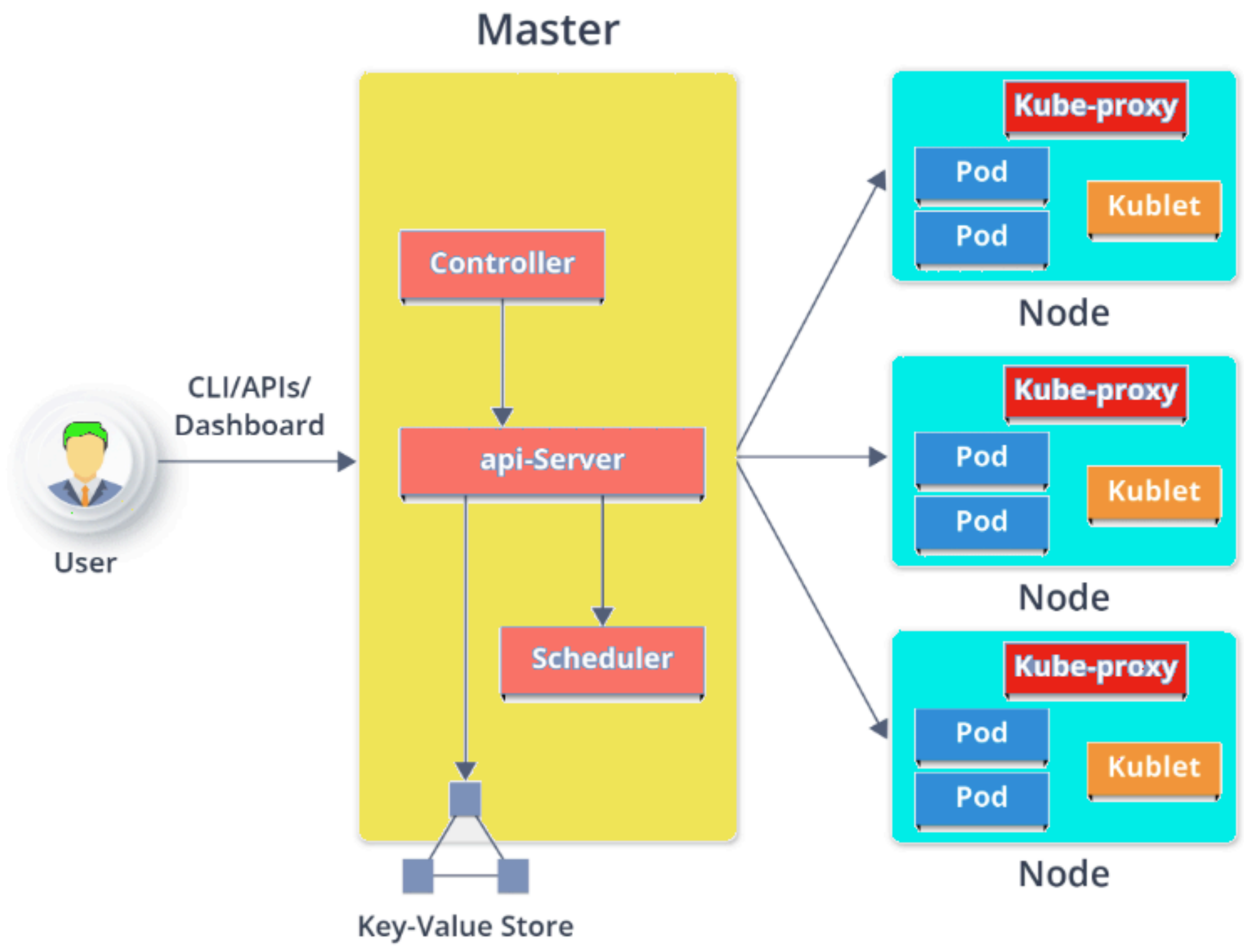
kubernetes

Kubernetes: Architecture

KUBERNETES : Architecture

- Kubernetes follow the Master - Slave(Worker) Node Architecture.
- **Master Node :** Responsible for the management of Kubernetes cluster. It is mainly the entry point for all administrative tasks.
- More than One Master Nodes be the there in Kubernetes Cluser.

KUBERNETES : Architecture



KUBERNETES : Architecture

- **Master Node:** Entry Point for All Administrative Tasks.
- Multiple Master Can be possible.
- In **Multi Master Node System**, Single Master node will be commanding Node for own workers and other Masters too.
- Main Master Node uses **etcd** to manage the Workers and Other Master Nodes.

- **API Server :** API server is the entry point for all the REST commands used to control the cluster.
- Resulting state of the cluster is stored in the **distributed key-value store**.

KUBERNETES : Architecture

- **Controller :** Regulates the Kubernetes cluster which manages the different non-terminating control loops.
- Performs lifecycle functions such as namespace creation, event garbage collection, node garbage collection, etc.
- Controller watches the desired state of the objects it manages and watches their current state through the API server.
- If the current state of the objects it manages does not meet the desired state, then the control loop takes corrective steps to make sure that the current state is the same as the desired state.

KUBERNETES : Architecture

- **Scheduler :** Regulates the tasks on slave nodes. It stores the resource usage information for each slave node.
- Schedules the work in the form of Pods and Services.

- **ETCD :** Distributed key-value store which stores the cluster state.
- Can be part of the Kubernetes Master, or, it can be configured externally.
- It's mainly used for shared configuration and service discovery.

KUBERNETES : Architecture

- **Worker Node :** It's a physical server or you can say a VM which runs the applications using Pods.
- Worker nodes contain all the necessary **services to manage** the **networking** between the containers, communicate with the master node, and assign resources to the scheduled containers.

- **Kubelet :** It is an agent which communicates with the Master node and executes on nodes or the worker nodes.
- Kubelet gets the configuration of a Pod from the API server and ensures that the described containers are up and running.

KUBERNETES : Architecture

- **Pods :** Is a group of one or more containers with shared storage/network, and a specification for how to run the containers.
- Pods run on nodes and run together as a logical unit.
- Share the Same Shared content and same IP but reach other Pods via LocalHost.
- Single Pod can Run on Multiple Machines and Single Machine can Run Multiple Pods.

KUBERNETES : Architecture

- **Kube-Proxy :** Kube-proxy runs on each node to deal with individual host sub-netting and ensure that the services are available to external parties.
- Kube-proxy acts as a network proxy and a load balancer for a service on a single worker node.
- It is the network proxy which runs on each worker node and listens to the API server for each Service endpoint creation/deletion.

Will see you in Next Lecture…

See you in next lecture …