There are mainly 4 problems to solve with the networking concepts.

* Container to container communication
* Pod to pod Communication
* Pod to service communication
* External to service Communication

Now, let me tell you how are the above problems are solved with Kubernetes Networking.

**Kubernetes Networking**

The communication between pods, services and external services to the ones in a cluster brings in the concept of Kubernetes networking.

So, for your better understanding let me divide the concepts into the following.

* Pods & Container Communication
* Services

**Pods & Container Communication**

Before I tell you how do pods communicate, let me introduce you what are pods?

**Pods**

Pods are basic units of Kubernetes applications, that consists of one or more containers allocated on the same host to share a network stack and other resources. So, this implies that all containers in a pod can reach other on a local host.

Now, let me brief you on how do these pods communicate?

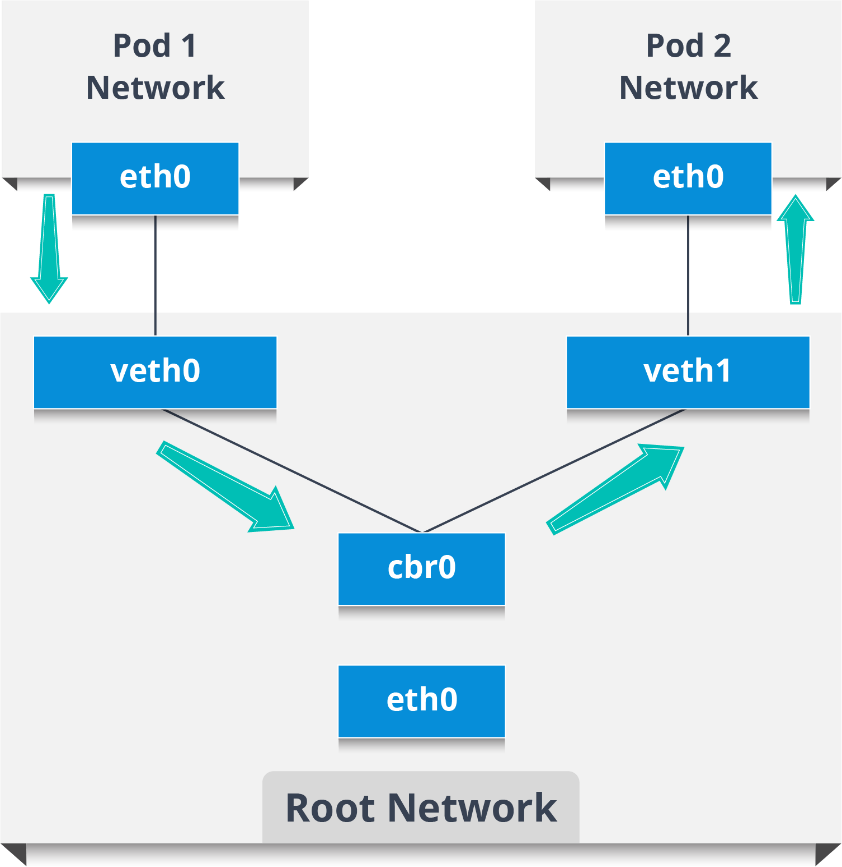
There are 2 types of communication. The**inter-node communication** and the **intra-node communication.**

So, let’s start with intra-node communication, but before that let me introduce to you the components of the pod network.

**Intra-node Pod Network**

Intra-node pod network is basically the communication between two different nodes on the same pod. Let me explain you with an example.

Assume a packet is going from pod1 to pod2.

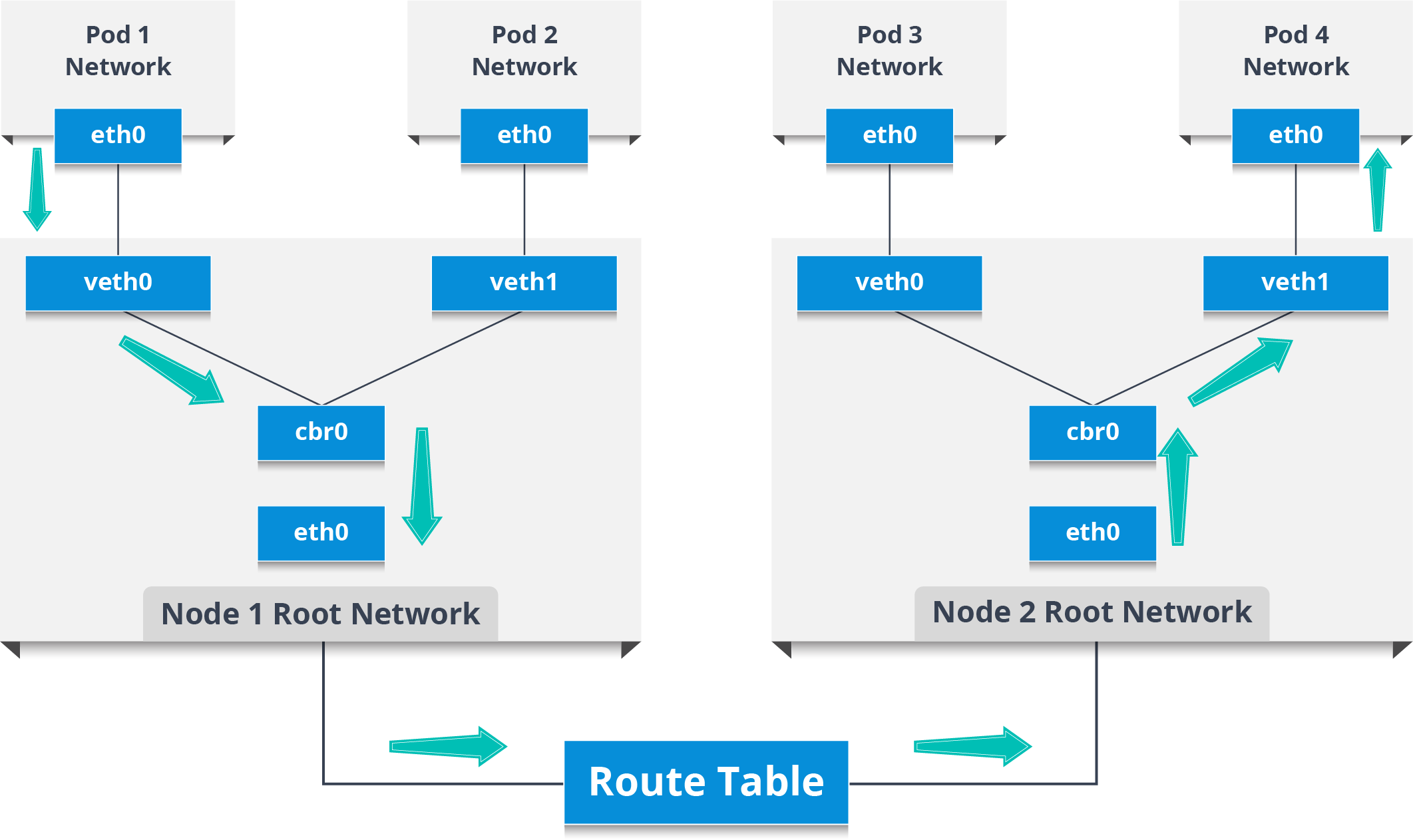


* The packet leaves Pod 1’s network at eth0 and enters the root network at veth0
* Then, the packet passes onto the Linux bridge(cbr0) which discovers the destination using an ARP request
* So, if veth1 has the IP, the bridge now knows where to forward the packet.

Now, similarly let me tell you about the inter-node pod communication.

**Inter-node pod network**

Consider two nodes having various network namespaces, network interfaces, and a Linux bridge.



Now, assume a packet travels from pod1 to a pod4 which is on a different node.

* The packet leaves the pod 1 network and enters the root network at veth0
* Then the packet passes on to the Linux bridge (cbr0) whose responsibility is to make an ARP request to find the destination.
* After the bridge realizes that this pod doesn’t have the destination address, the packet comes back to the main network interface eth0.
* The packet now leaves the node 1 to find it’s destination on the other node and enters the route table who routes the packet to the node whose CIDR block contains the pod4.
* So, now the packet reaches node2 and then the bridge takes the packet which makes an ARP request to find out that the IP belonging to veth0.
* Finally, the packet crosses the pipe-pair and reaches pod4.

So, that’s how pods communicate with each other. Now, lets’ move on and see how services help in the communication of pods.

So, what do you think the services are?

**[Kubernetes Certification Training](https://www.edureka.co/kubernetes-certification" \t "_blank)**

* *[Instructor-led Sessions](https://www.edureka.co/kubernetes-certification" \t "_blank)*
* *[Real-life Case Studies](https://www.edureka.co/kubernetes-certification" \t "_blank)*
* *[Assignments](https://www.edureka.co/kubernetes-certification" \t "_blank)*
* *[Lifetime Access](https://www.edureka.co/kubernetes-certification" \t "_blank)*

[Explore Curriculum](https://www.edureka.co/kubernetes-certification" \t "_blank)

**Services**

Basically, services are a type of resource that configures a proxy to forward the requests to a set of pods, which will receive traffic & is determined by the selector. Once the service is created it has an assigned IP address which will accept requests on the port.

Now, there are various service types that give you the option for exposing a service outside of your cluster IP address.

**Types of Services**

There are mainly 4 types of services.

**ClusterIP:** This is the default service type which exposes the service on a cluster-internal IP by making the service only reachable within the cluster.

**NodePort:** This exposes the service on each Node’s IP at a static port. Since, a **ClusterIP**service, to which the NodePort service will route, is automatically created. We can contact the NodePort service outside the cluster.

**LoadBalancer:**This is the service type which exposes the service externally using a cloud provider’s load balancer. So, the NodePort and ClusterIP services, to which the external load balancer will route, are automatically created.

**ExternalName**: This service type maps the service to the contents of the **externalName** field by returning a  **name** record with its value.