**The Kubernetes network model:**

1. Every Pod in a cluster gets its own unique cluster-wide IP address.
2. This means you do not need to explicitly create links between Pods and you almost never need to deal with mapping container ports to host ports.
3. This creates a clean, backwards-compatible model where Pods can be treated much like VMs or physical hosts from the perspectives of port allocation, naming, service discovery, load balancing, application configuration, and migration.
4. Kubernetes imposes the following fundamental requirements on any networking implementation

○ pods can communicate with all other pods on any other node without NAT

○ agents on a node (e.g. system daemons, kubelet) can communicate with all pods on that node

Note:

* For those platforms that support Pods running in the host network (e.g. Linux), when pods are attached to the host network of a node they can still communicate with all pods on all nodes without NAT.
* This model is not only less complex overall, but it is principally compatible with the desire for Kubernetes to enable low-friction porting of apps from VMs to containers.
* If your job previously ran in a VM, your VM had an IP and could talk to other VMs in your project.

**This is the same basic model.**

1. Kubernetes IP addresses exist at the Pod scope - containers within a Pod share their network namespaces - including their IP address and MAC address.
2. This means that containers within a Pod can all reach each other's ports on localhost.
3. This also means that containers within a Pod must coordinate port usage, but this is no different from processes in a VM. This is called the "IP-per-pod" model.
4. How this is implemented is a detail of the particular container runtime in use.
5. It is possible to request ports on the Node itself which forward to your Pod (called host ports), but this is a very niche operation.
6. How that forwarding is implemented is also a detail of the container runtime.
7. The Pod itself is blind to the existence or non-existence of host ports.

**Kubernetes networking addresses four concerns:**

• Containers within a Pod use networking to communicate via loopback.

• Cluster networking provides communication between different Pods.

• The Service resource lets you expose an application running in Pods to be reachable from outside your cluster

• You can also use Services to publish services only for consumption inside your cluster