

## # Kubernetes Networking – Reference

### ## Networking Model

Kubernetes uses a flat networking model where every Pod receives its own unique IP address. Pods can communicate with each other across nodes without NAT (Network Address Translation).

This model simplifies service discovery and avoids port conflicts between applications.

### ## CNI Plugins (Container Network Interface)

CNI plugins handle Pod-to-Pod networking at the infrastructure level.

- Calico: supports network policy enforcement and BGP-based routing
- Cilium: uses eBPF for high-performance, kernel-level packet processing
- Flannel: simple overlay network using VXLAN, minimal policy support
- Weave: mesh overlay with built-in encryption

### ## Kubernetes Services

Services expose Pods over a stable virtual IP.

- ClusterIP: internal-only access within the cluster
- NodePort: exposes service on a static port on each node
- LoadBalancer: provisions an external cloud load balancer
- ExternalName: maps a service to an external DNS name

### ## kube-proxy Modes

kube-proxy handles service-to-pod routing rules on each node.

- iptables mode: default, uses kernel netfilter rules
- IPVS mode: better performance at scale using Linux Virtual Server
- eBPF replacement (Cilium): bypasses kube-proxy entirely for lower latency

### ## Ingress vs Gateway API

Ingress manages external HTTP/HTTPS routing to services at Layer 7.

Gateway API is the newer standard that supports multi-tenancy, traffic splitting, and cleaner separation between infrastructure and application teams.

### ## Network Policies

Network Policies control traffic flow between Pods using selectors.

- podSelector: targets specific Pods by label
- namespaceSelector: targets Pods in specific namespaces
- ingress rules: control inbound traffic to selected Pods
- egress rules: control outbound traffic from selected Pods

Network Policies require a CNI plugin that supports enforcement (e.g., Calico, Cilium).

### ## DNS in Kubernetes

CoreDNS is the default DNS server in Kubernetes, replacing the older kube-dns.

It resolves service names within the cluster (e.g., my-service.my-namespace.svc.cluster.local).

stubDomains allow forwarding specific DNS zones to custom resolvers.

## ## Service Mesh

Service meshes like Istio and Linkerd inject sidecar proxies alongside each Pod. These proxies intercept all inbound and outbound traffic to provide observability, mutual TLS, retries, and traffic management without changing application code.