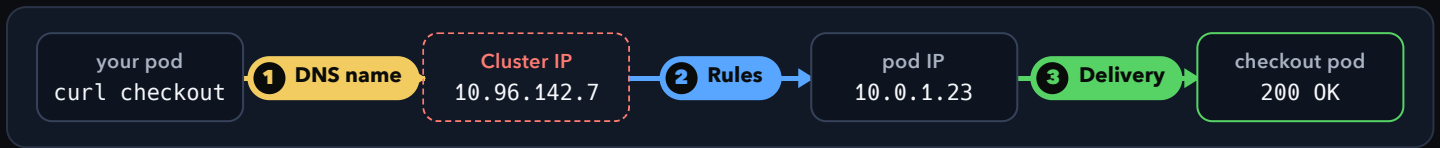


How Kubernetes Actually Routes Traffic

A **Kubernetes pod** sends a request with curl to **checkout**, another app in the same cluster. Three steps get it there.



1 The DNS name The name checkout turns into an IP address.

The cluster's phone book

- A DNS server, usually **CoreDNS**, running in a pod
- A DNS name in, an IP address out

The Cluster IP

- The address of checkout's **Service**, reachable only inside the cluster
- **On no node and no pod: nothing at it can answer**

kube-proxy writes the rules

- One copy per node; watches the Kubernetes API
- Writes rules into the kernel: *this address → one of these pods*
- Your request never passes through it

2 The rules On your own node, the kernel swaps the address.

iptables mode: one long list

- kube-proxy's default mode
- The kernel reads the list from the top until one entry matches
- **Catch: tens of thousands of Services make it slow**

The swap: DNAT KEY MOMENT

- TO ~~10.96.142.7~~ → 10.0.1.23
- A ready pod, picked at random
 - Done in your own node's kernel
 - D is for destination

The conntrack notebook

- Remembers the swap: the whole connection goes to the same pod
- Swaps the reply's sender back, so curl never notices

3 The delivery The packet crosses to the chosen pod's node.

The network plugin

- An add-on such as **Calico** or **Cilium**
- Gives every pod an IP address and a network card, made in software

A tunnel between nodes

- Network knows only node IPs: a second envelope, to the other node
- Knows pod IPs too: no tunnel

Delivered, then the reply

- The other node unwraps it and hands it to the checkout pod
- The reply takes the same way back

+ Two fixes for the slow list And which one to use.

Fix 1: nftables mode

- Looks the Cluster IP up directly, like a book's index
- Updates only what changed
- Fully supported; planned to become the default

Fix 2: eBPF with Cilium

- Cilium's **kube-proxy replacement**, off by default if you install Cilium yourself
- Picks the pod as the connection opens: no DNAT, no swap back

Which to use

- A few hundred Services: any of the three
- Tens of thousands: **nftables mode** (kube-proxy 1.36+), or Cilium's replacement if Cilium is your network plugin

CHECK YOUR MODE (ON ANY NODE)

```
$ curl -s localhost:10249/proxyMode
iptables
```

Then set the mode on purpose, so an upgrade doesn't change it for you.

SET IT ON PURPOSE (KUBEADM CLUSTERS)

```
$ kubectl -n kube-system edit configmap kube-proxy
# in config.conf: mode: "iptables" (or "nftables")
```

```
$ kubectl -n kube-system rollout restart daemonset kube-proxy
```

nftables needs Linux 5.13+ on every node. The restart can briefly interrupt traffic: try it on a test cluster first.