Perfect 👍 — here’s a **simple and practical test setup** to verify **communication between pods on different nodes** in Kubernetes.

**🧩 Step 1: Create a Test Deployment**

Let’s create two pods running busybox — a lightweight Linux container useful for network testing.

Create a file named **network-test.yaml** 👇

apiVersion: v1

kind: Pod

metadata:

name: pod-a

labels:

app: nettest

spec:

containers:

- name: busybox

image: busybox

command: ["sleep", "3600"]

---

apiVersion: v1

kind: Pod

metadata:

name: pod-b

labels:

app: nettest

spec:

containers:

- name: busybox

image: busybox

command: ["sleep", "3600"]

Now apply:

kubectl apply -f network-test.yaml

**🧩 Step 2: Check Where Each Pod Is Running**

Run:

kubectl get pods -o wide

Example output:

NAME READY STATUS NODE IP AGE

pod-a 1/1 Running workernode1 10.244.1.7 2m

pod-b 1/1 Running workernode2 10.244.2.5 2m

✅ Perfect — pods are on **different nodes**.

**🧩 Step 3: Test Pod-to-Pod Communication**

Now, **enter pod-a**:

kubectl exec -it pod-a -- sh

From inside pod-a, run:

ping <pod-b-ip>

Example:

ping 10.244.2.5

You should see something like:

PING 10.244.2.5 (10.244.2.5): 56 data bytes

64 bytes from 10.244.2.5: seq=0 ttl=62 time=0.75 ms

✅ If it works → pod-to-pod communication (cross-node) is working properly.

❌ If it fails → there’s a **CNI issue** (Calico/Flannel/etc. not configured properly).

**🧩 Step 4: Test Connectivity from Pod to Node**

Still inside pod-a, run:

ping <node-ip>

If you get responses → Pod → Node communication is fine.

**🧩 Step 5: Clean Up**

Once testing is done:

kubectl delete -f network-test.yaml

**🧠 Summary**

| **Test** | **Command** | **Result** |
| --- | --- | --- |
| Pod placement | kubectl get pods -o wide | See which node hosts which pod |
| Pod → Pod | ping <pod-ip> | Tests CNI network |
| Pod → Node | ping <node-ip> | Verifies node access |
| Cleanup | kubectl delete -f network-test.yaml | Removes test pods |