**Challenges and Solutions in Kubernetes Networking and Gateway Configuration on OKE**

**Introduction**

Managing Kubernetes networking in **Oracle Kubernetes Engine (OKE)** involves more than just deploying workloads. It requires configuring **gateways, route tables, and subnets** so that clusters can communicate securely — both internally and externally. In our journey, we faced several challenges while setting up proper routing for the **API endpoint, worker nodes, and load balancers**, and here’s how we solved them step by step.

**1. Challenge: Shared Route Tables Exposing Nodes**

Initially, all three OKE subnets were using the **same public route table**:

* **API Endpoint Subnet**
* **Worker Node Subnet**
* **Service Load Balancer Subnet**

This route table had a **default route (0.0.0.0/0) → Internet Gateway (IGW)**.

**Problem:**

* The **worker nodes** were directly exposed to the internet, which is insecure and against best practices.
* When we tried to move the API endpoint subnet to NAT, we **lost kubectl access from VS Code** because the Kubernetes API must stay publicly reachable.

**2. Solution: Introduce a NAT Gateway**

To fix this, we created a **dedicated NAT Gateway** for the worker nodes. NAT allows nodes to make outbound connections (e.g., pull images, reach Let’s Encrypt, fetch updates) without being directly exposed.

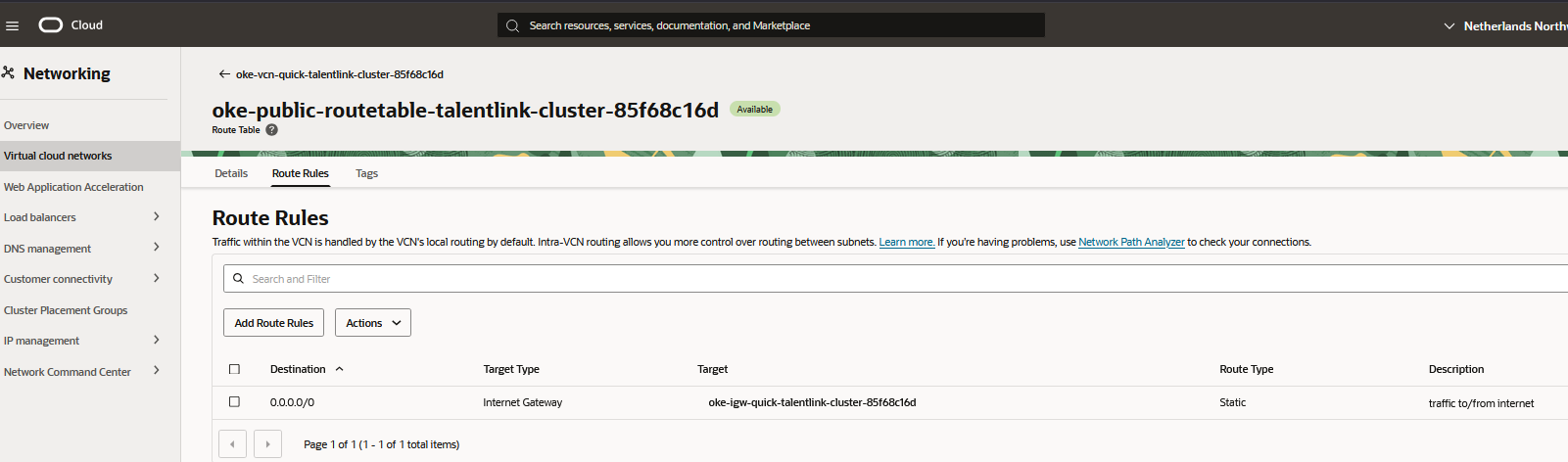
**OCI Console Steps:**

* Go to **Networking → Virtual Cloud Networks → [VCN] → NAT Gateways → Create NAT Gateway**
* Name: oke-nat-gateway

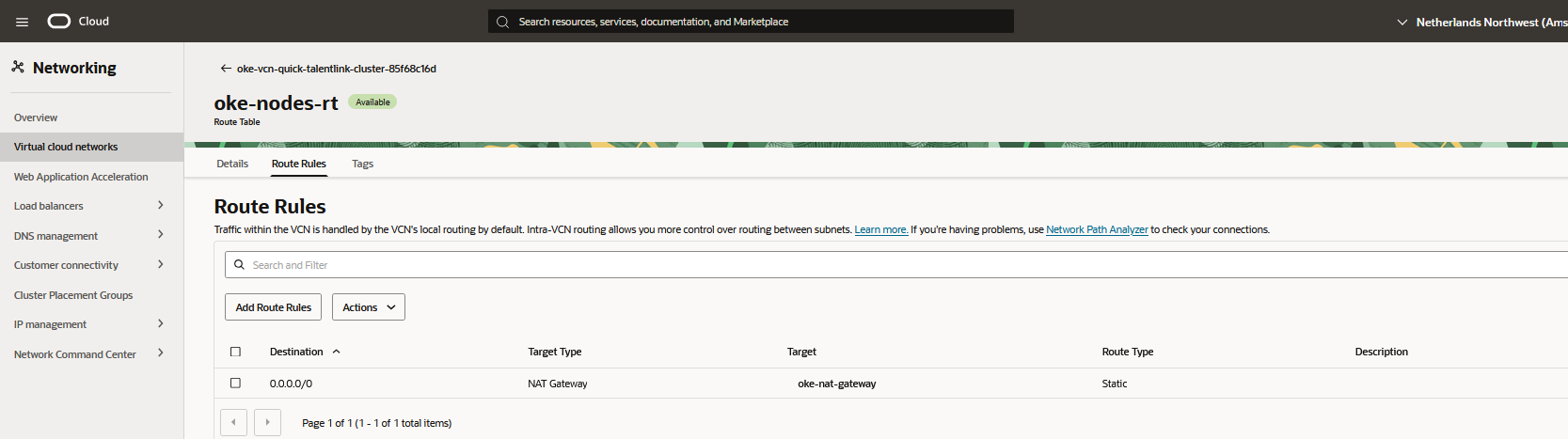
**3. Solution: Create Dedicated Route Tables**

We separated traffic into **two custom route tables**:

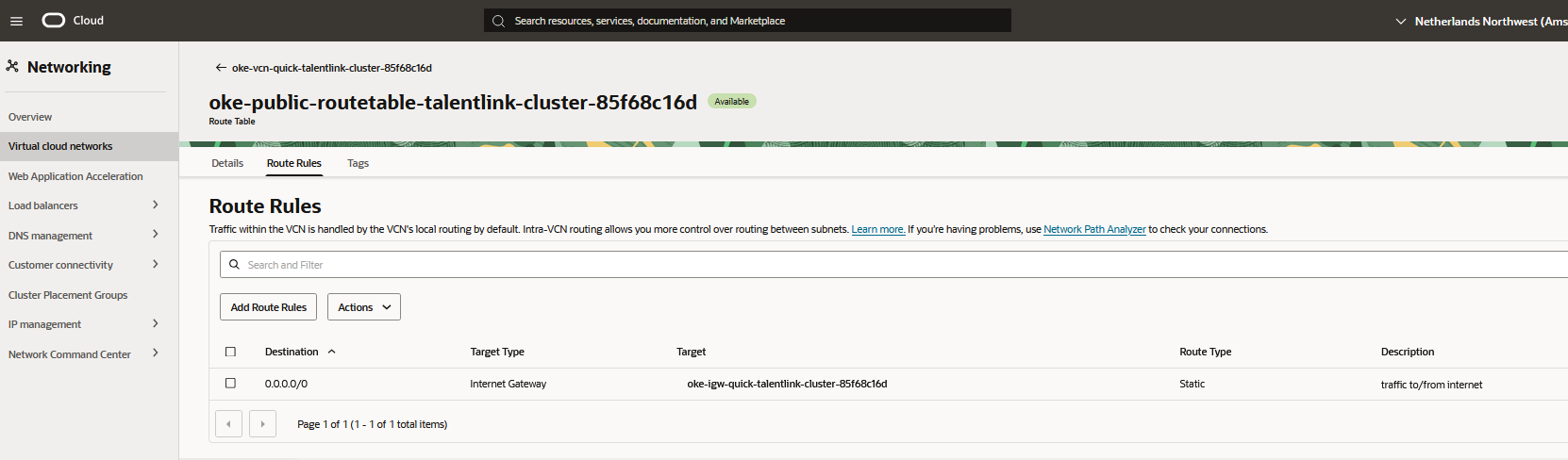
1. **For API Endpoint Subnet** → must always use IGW



1. **For Worker Node Subnet** → must use NAT Gateway



1. **For Service LB Subnet** → kept on IGW (public ingress/egress)



**4. Challenge: API Endpoint Route Table Is Managed**

When we tried to edit the **API endpoint subnet**, we discovered the **Edit → Change Route Table** option was missing in OCI Console.

**Why?**

* In OKE, the API endpoint subnet is **managed by Oracle** and must remain on IGW.
* Attempting to put it behind NAT breaks kubectl access.

**Resolution:**

We **left the API endpoint subnet on IGW** and only reassigned the **Node Subnet** to use the NAT route table.

**5. Solution: Reassign Subnets**

Using the OCI Console:

* **Node Subnet** → changed route table to oke-nodes-rt (NAT Gateway)
* **API Endpoint Subnet** → kept on oke-public-routetable (IGW)
* **Service LB Subnet** → stayed on oke-public-routetable (IGW)

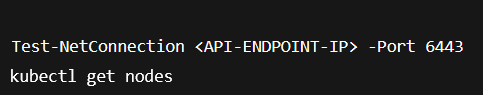
**6. Challenge: Verifying Connectivity After Changes**

After reassigning subnets, we needed to confirm both:

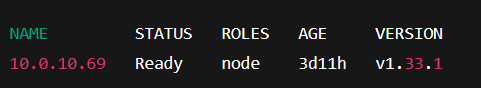
* **kubectl access** from VS Code still worked
* **worker nodes** could still reach the internet

**Commands used:**

**API Reachability (from local PowerShell)**



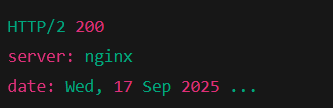
Output:



This proved the API endpoint was still accessible.

**Node Internet Reachability (from inside the cluster)**

We launched a temporary pod to test outbound access:

  
  
**Result:  
**

**This confirmed the nodes could successfully reach external services via the NAT gateway.**

**7. Final State**

* **API Endpoint Subnet → Internet Gateway**
* **Worker Node Subnet → NAT Gateway**
* **Service LB Subnet → Internet Gateway**

**This setup achieved the balance of security + functionality:**

* **API endpoint is reachable for cluster management**
* **Worker nodes are hidden but can access the internet outbound**
* **Load balancers are publicly accessible**

**Conclusion**

**The main challenge was understanding which subnets need IGW vs NAT and avoiding the pitfall of moving the API endpoint behind NAT (which broke kubectl access).**

**By carefully splitting routing rules:**

* **API endpoint via IGW**
* **Worker nodes via NAT**
* **Load balancers via IGW**

**…and validating connectivity with PowerShell and kubectl commands, we established a secure and functional networking design for OKE.**

**This experience reinforced the importance of network isolation in Kubernetes and highlighted how OCI-managed subnets (like API endpoint) behave differently from user-managed subnets.**