# Networking

* Create a Pod for testing networking. We will get access to its terminal and that Pod will be deleted after exiting it:
  + kubectl run dns-test --image=busybox:1.28 --restart=Never -it --rm -- /bin/sh

From the Pod’s terminal try to resolve DNS names:

* + nslookup kubernetes.default – DNS name for API server
  + nslookup google.com – check a public domain
  + nslookup <service-name>.<namespace> - domain name for some Kubernetes Service

## CoreDNS

* Check CoreDNS logs
* Check firewall rules using iptables
* Check CoreDNS configMap, especially section with the ‘forward’ keyword
* Check the resolv.conf file of a Pod.
* kubectl rollout restart deployment coredns -n kube-system – restart a coredns deployment.

# Pods

* kubectl get pod <pod-name> -n <namespace> -o yaml – Get the Pod specification in YAML format.
* kubectl logs -n <namespace\_name> <pod-name> - check logs of a Pod from a given namespace

# Roles

* kubectl get rolebinding,clusterrolebinding --all-namespaces -o wide | grep <name> – get roles assigned to all the users, groups and Service Accounts. ‘grep’ filters outputs to find a specific string.

# Setting up Kubernetes

Here are usefull tools for debugging.

* crictl ps -a – list the running and stopped containers.
* Crictl logs <containerID> - check logs for a given container
* ss -tuln | grep 6443 – check processes listening on the 6443 port
* ps aux | grep kube-apiserver – check processes in which the ‘kube-apiserver’ string appears.
* journalctl -u kubelet -f – view logs from the kubelet process
* kubectl get pods -n kube-system – Check pods in the kube-system namespace
* telnet 10.0.1.4 6443 – try to connect over TCP to the server with the 10.0.1.4 IP address over the 6443 port.