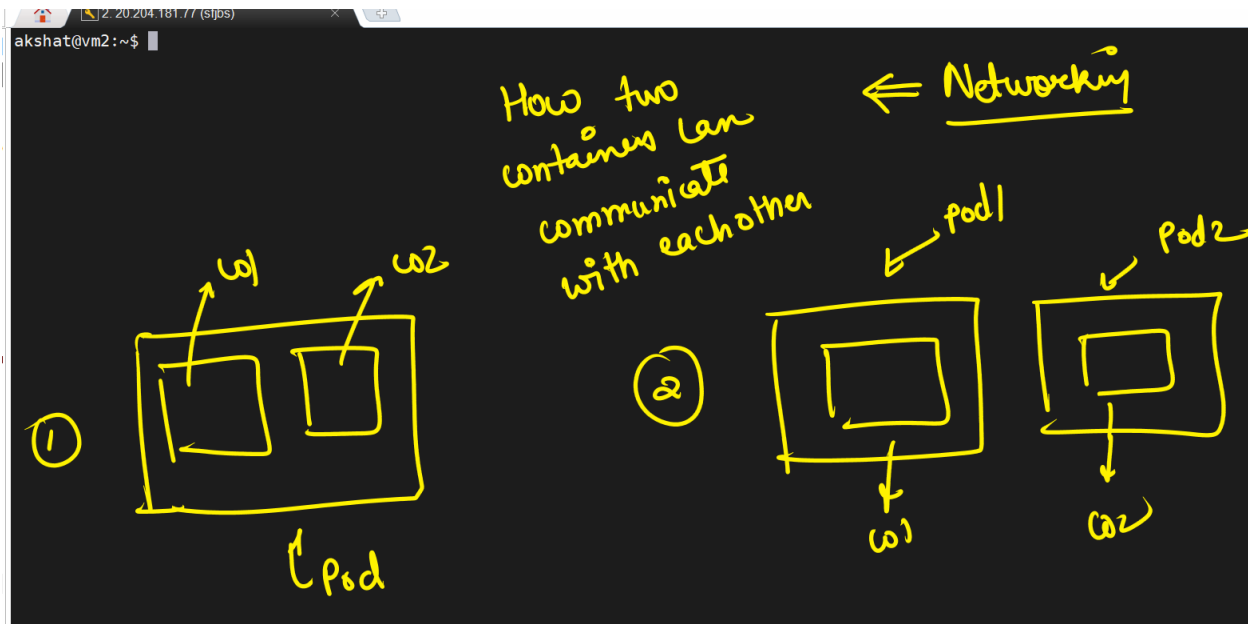
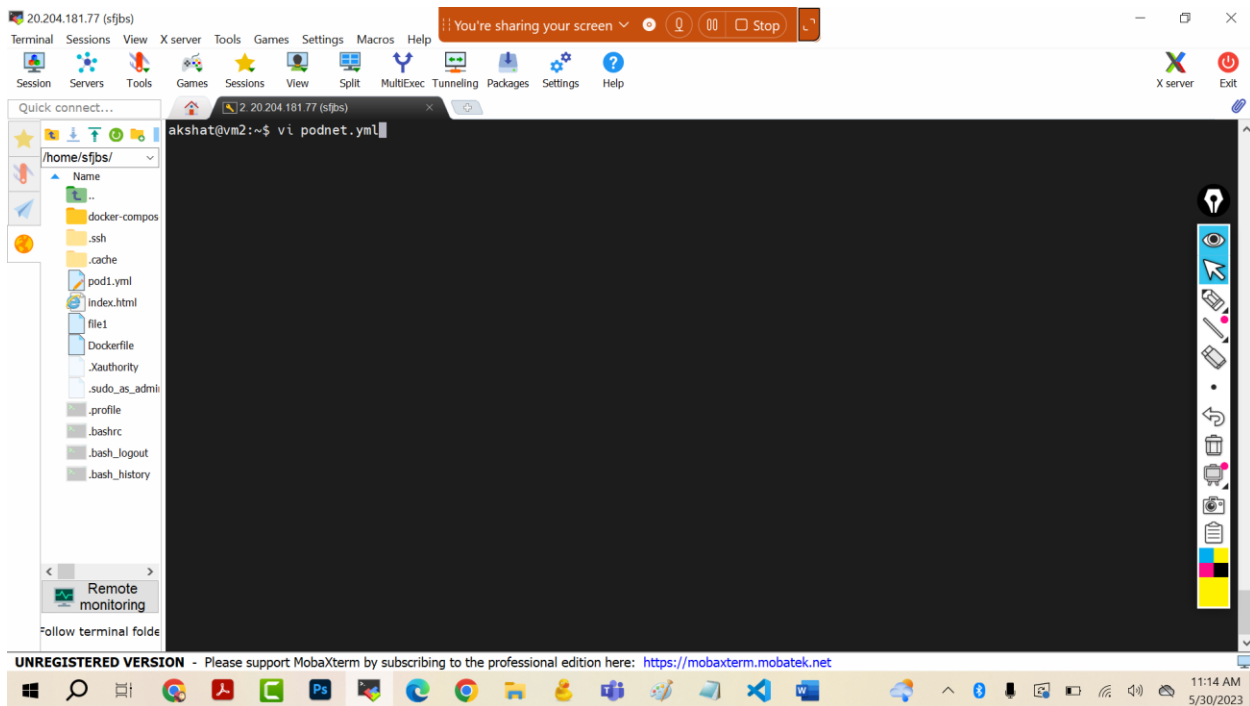


Networking and eni



Create two container in a single pod and check if they are able to communicate with each other or not



We will get inside the pod and in container c00 and check if we are able to access web server httpd via that or not

```
kind: Pod
apiVersion: v1
metadata:
  name: testpod1
spec:
  containers:
    - name: c00
      image: ubuntu
      command: ["/bin/bash", "-c", "while true; do echo Hello akshat; sleep 5; done"]
    - name: c01
      image: httpd
      ports:
        - containerPort: 80
```

```
akshat@vm2:~$ vi podnet.yml
akshat@vm2:~$ kubectl apply -f podnet.yml
pod/testpod1 created
akshat@vm2:~$
```

```
akshat@vm2:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
testpod1      2/2     Running   0           13s
akshat@vm2:~$
```

Now we will enter inside the c00 of pod testpod1 and check if we are able to access the files or not

```
akshat@vm2:~$ kubectl exec testpod1 -it -c c00 -- /bin/bash
```

-it : interactive terminal

-c : container

/bin/bash : bash terminal

```
apt update -y
```

```
apt install curl -y
```

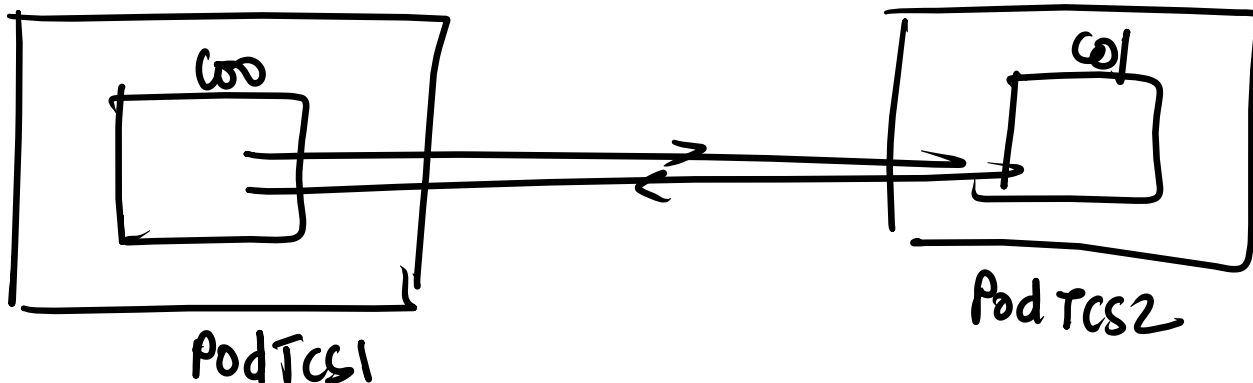
```
curl localhost:80
```

```
root@testpod1:/# curl localhost:80
<html><body><h1>It works!</h1></body></html>
root@testpod1:/#
```

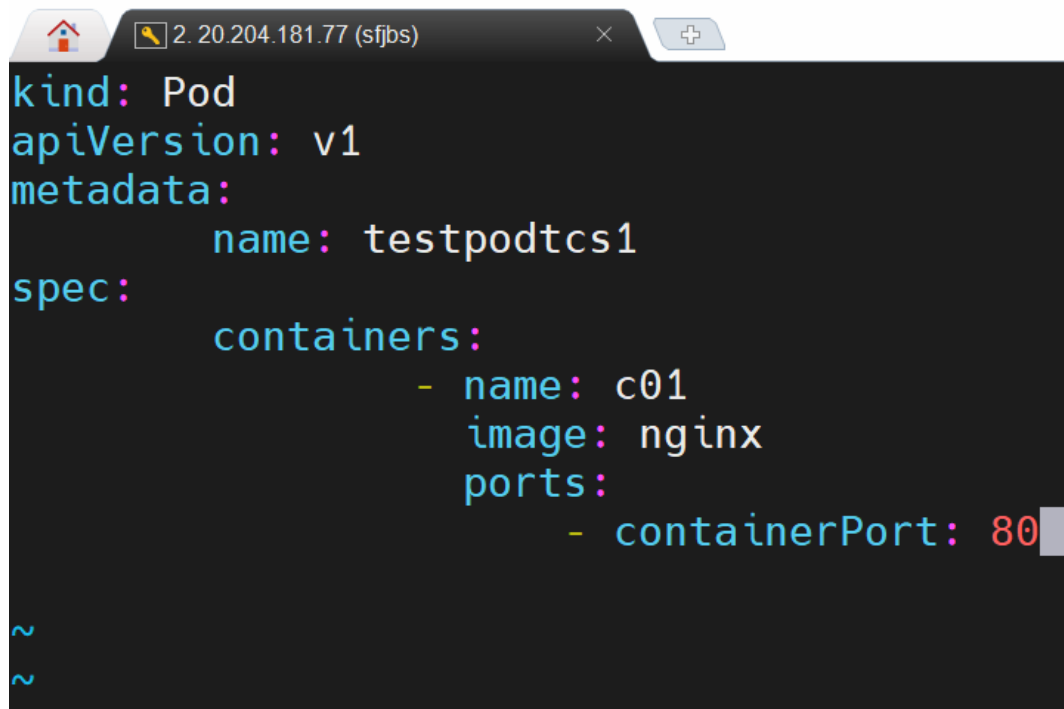
```
exit
```

Two containers in the same pod can communicate with each other directly

TWO PODS HAVING TWO DIFFERENT CONTAINERS



```
akshat@vm2:~$ vi podtcs1.yml
```

A screenshot of a web browser window with a single tab titled '2. 20.204.181.77 (sfjbs)'. The browser displays a YAML configuration for a Kubernetes Pod. The content is as follows:

```
kind: Pod
apiVersion: v1
metadata:
  name: testpodtcs1
spec:
  containers:
    - name: c01
      image: nginx
      ports:
        - containerPort: 80
```

At the bottom of the page, there are two tilde (~) symbols.

```
akshat@vm2:~$ kubectl apply -f podtcs1.yml
pod/testpodtcs1 created
akshat@vm2:~$
```

Now lets create another pod :

```
vi testpodtcs2.yml
```

```
kind: Pod
apiVersion: v1
metadata:
  name: testpodtcs2
spec:
  containers:
  - name: c02
    image: httpd
    ports:
    - containerPort: 80
```

```
akshat@vm2:~$ kubectl apply -f testpodtcs.yml
pod/testpodtcs2 created
akshat@vm2:~$
```

```
akshat@vm2:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
testpod1      2/2     Running   0           11m
testpodtcs1   1/1     Running   0           2m28s
testpodtcs2   1/1     Running   0           20s
akshat@vm2:~$
```

```
akshat@vm2:~$ kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP            NODE   NOMINATED NODE   READINESS GATES
testpod1      2/2     Running   0           13m   10.244.0.20   minikube   <none>           <none>
testpodtcs1   1/1     Running   0           3m58s  10.244.0.21   minikube   <none>           <none>
testpodtcs2   1/1     Running   0           110s   10.244.0.22   minikube   <none>           <none>

akshat@vm2:~$ kubectl exec testpodtcs1 -it -- /bin/bash
root@testpodtcs1:/# curl 10.244.0.22
```

Handwritten blue notes:

- ip add^s of testpodtcs2
- getting inside testpodtcs1

```
akshat@vm2:~$ kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP            NODE   NOMINATED NODE   READINESS GATES
testpod1      2/2     Running   0           13m   10.244.0.20   minikube   <none>           <none>
testpodtcs1   1/1     Running   0           3m58s  10.244.0.21   minikube   <none>           <none>
testpodtcs2   1/1     Running   0           110s   10.244.0.22   minikube   <none>           <none>

akshat@vm2:~$ kubectl exec testpodtcs1 -it -- /bin/bash
root@testpodtcs1:/# curl 10.244.0.22
<html><body><h1>It works!</h1></body></html>
root@testpodtcs1:/#
```

File explorer contents (left):

- /home/sfjbs/
- docker-compose
- .ssh
- .cache
- pod1.yml
- index.html
- file1
- Dockerfile
- .Xauthority
- .sudo_as_admin
- .profile
- .bashrc
- .bash_logout
- .bash_history

Remote monitoring: take screenshot (ctrl + shift + printscreen)

exit

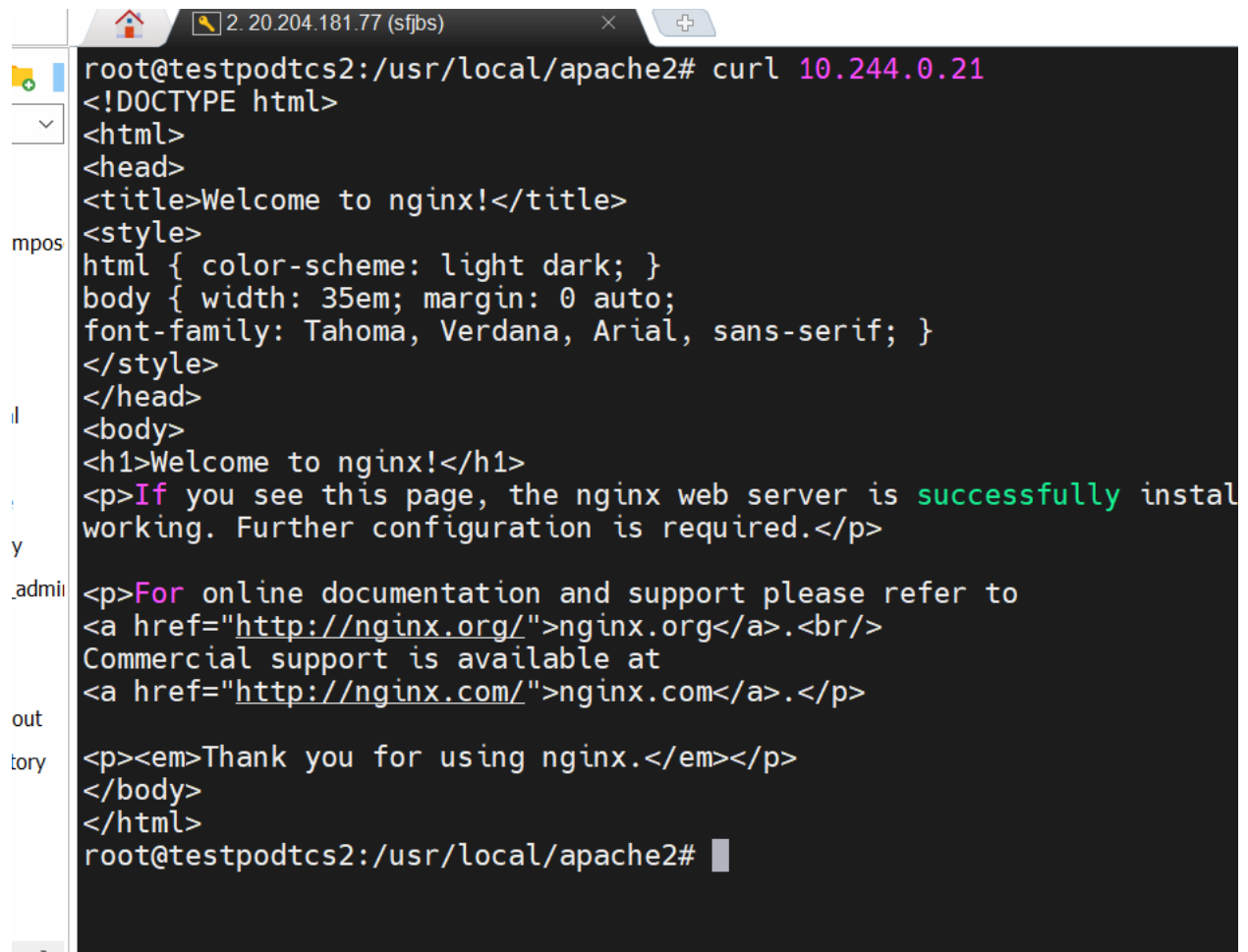
now lets get inside the testpodtcs2 and try to access the container webserver of testpodtcs1

```
kubect exec testpodtcs2 -it -- /bin/bash
```

```
apt update -y
```

```
apt install curl -y
```

curl <<ip address of testpodtcs1>>



```
root@testpodtcs2:/usr/local/apache2# curl 10.244.0.21
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed
and ready to use. The nginx web server is a powerful, high-performance
web server that can handle a large number of concurrent connections.
For online documentation and support please refer to
http://nginx.org/.<br/>
Commercial support is available at
http://nginx.com/.</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@testpodtcs2:/usr/local/apache2#
```

exit

It proves that within the cluster the containers inside the pod can communicate with each other.