

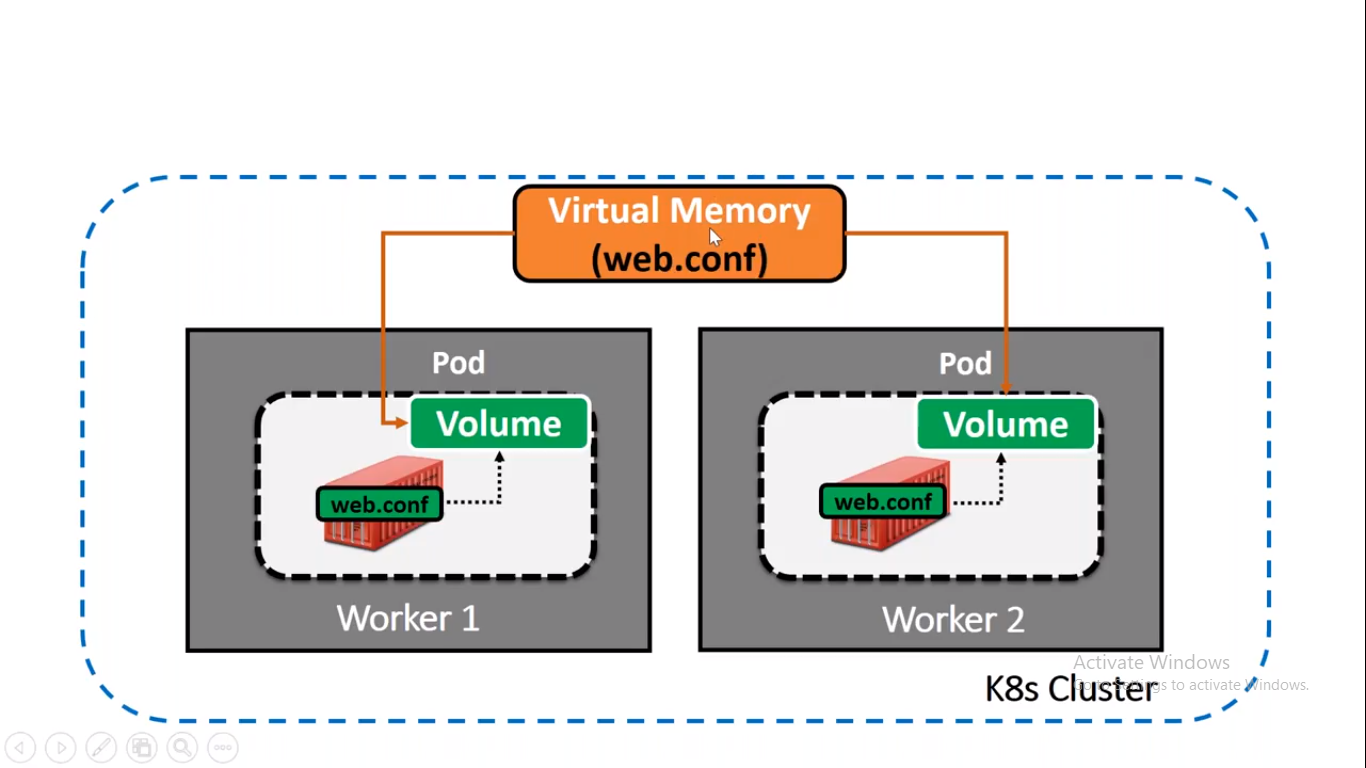
According to Docker Basics as we know that whenever Container is getting created its required base images and other layer like the file to read the database configuration and to connect it.

SO to do that we will be having one file called as Web.conf file and it will contains all details of database configuration by this we will create our product image and deploy in one environment.

I.e. in Dev Environment and same image we will be giving to the QA team to create the product so again there it will be using same file to read the database configuration details and connect to DEV Database.

But this is not correct. and similarly it will go to other environment as well like staging and Production ENV and it will try to connect the Dev database.

So to overcome from it. we need to go for volume concepts but again if we go to the Persistent volument that will be very expensive.



Here we will create a Virtual Memory and it wil be available in all env cluster.

And each cluster specific web.conf file should be available.

So that whenever Pod is created which has Volume mapped to it.

So that it will connect to the Virtual Memory and get the database details and connect to the db and this way we can pass the smog test.

This web.conf will be passed to the container during run time from Virtual Memory.

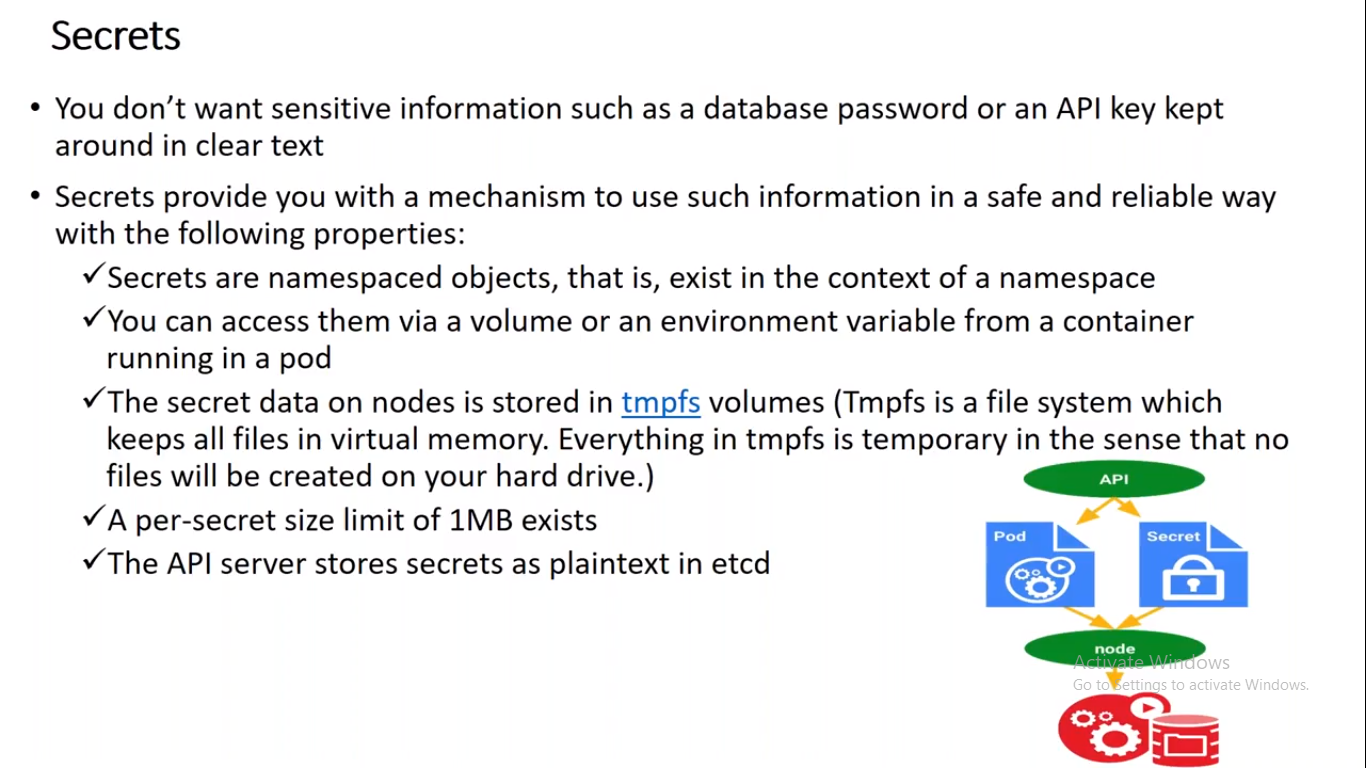
Here if we gor AWS EBS that too for one file that is very expensive.

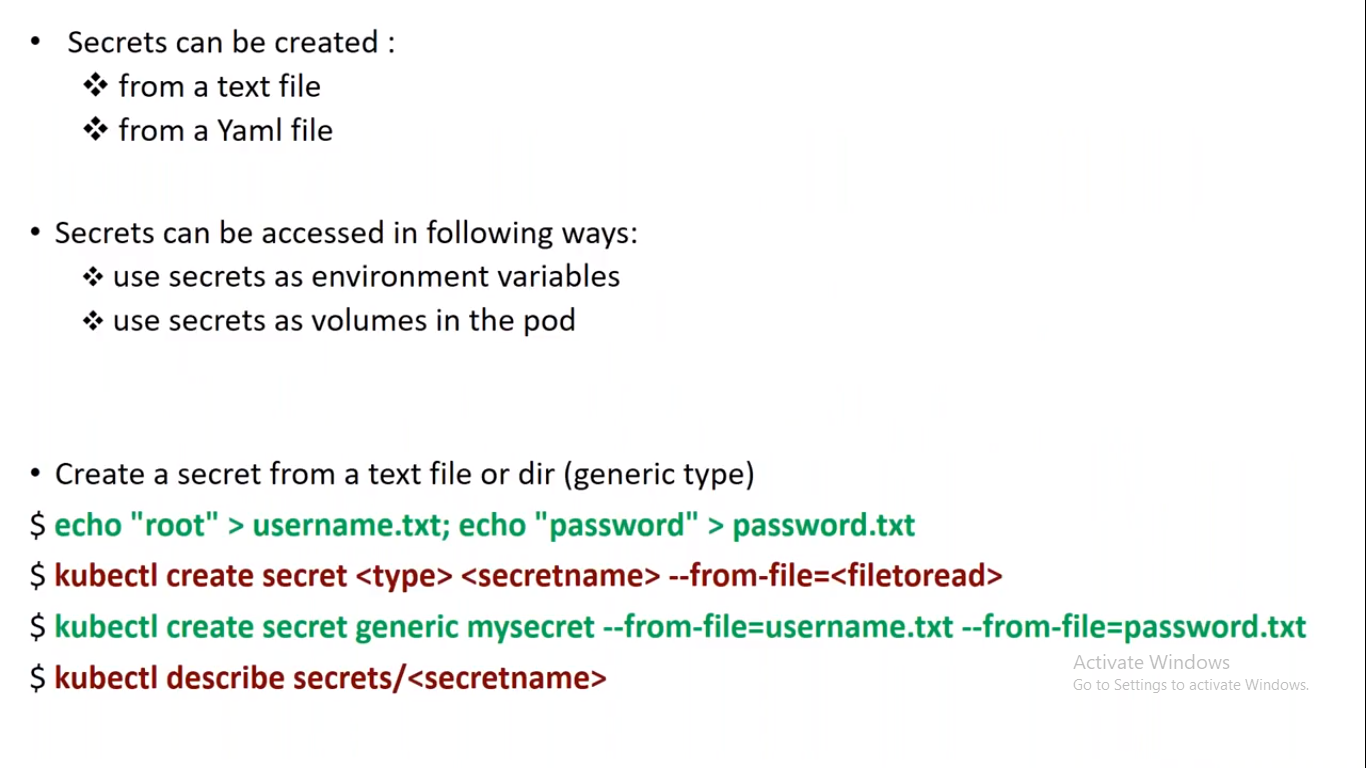
To achieve it we have two ways.

1. Secret
2. Config Map

Both are similar in functionality wise but difference is that when we want some configuration file to start the application then we will go for **ConfigMap**

**But** for some application we requires some sensitive details like password, private keys and all that are secure thing so for that we will be using **Secrets**

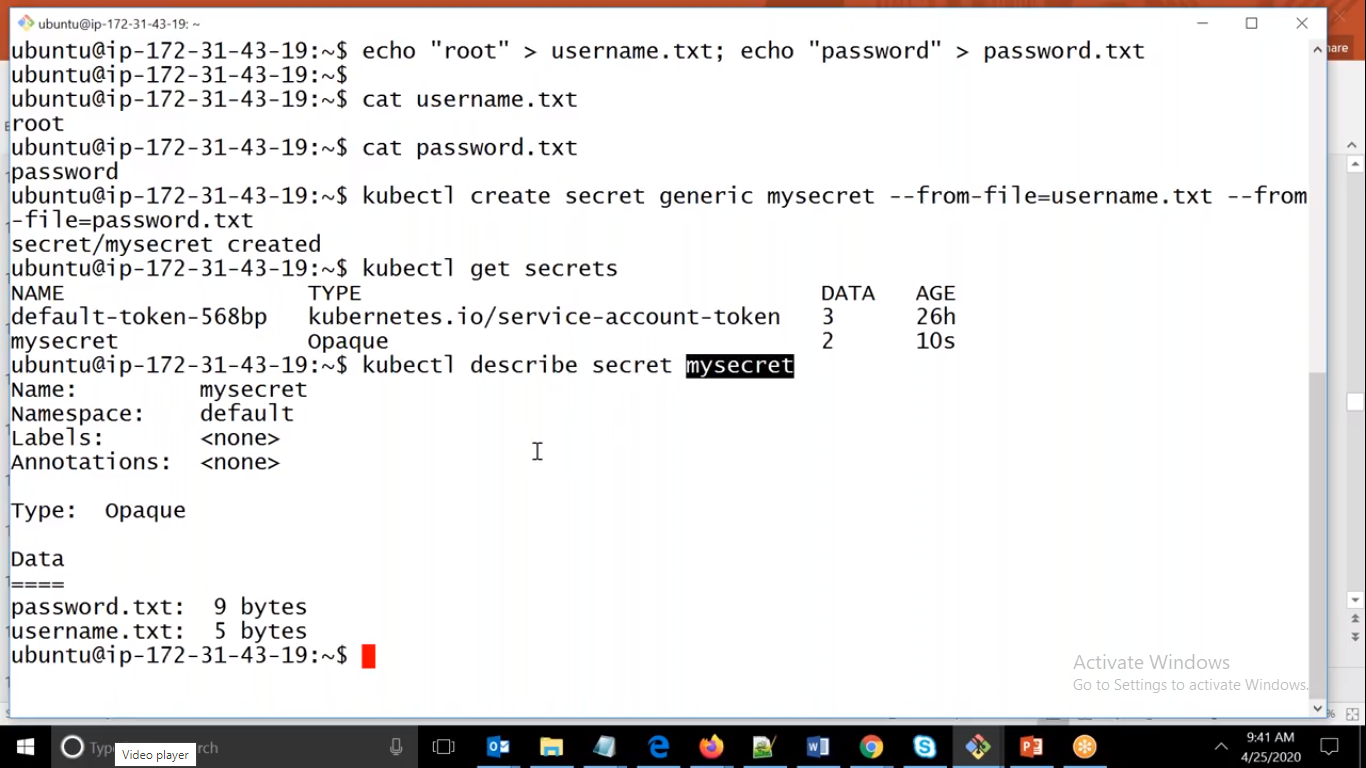


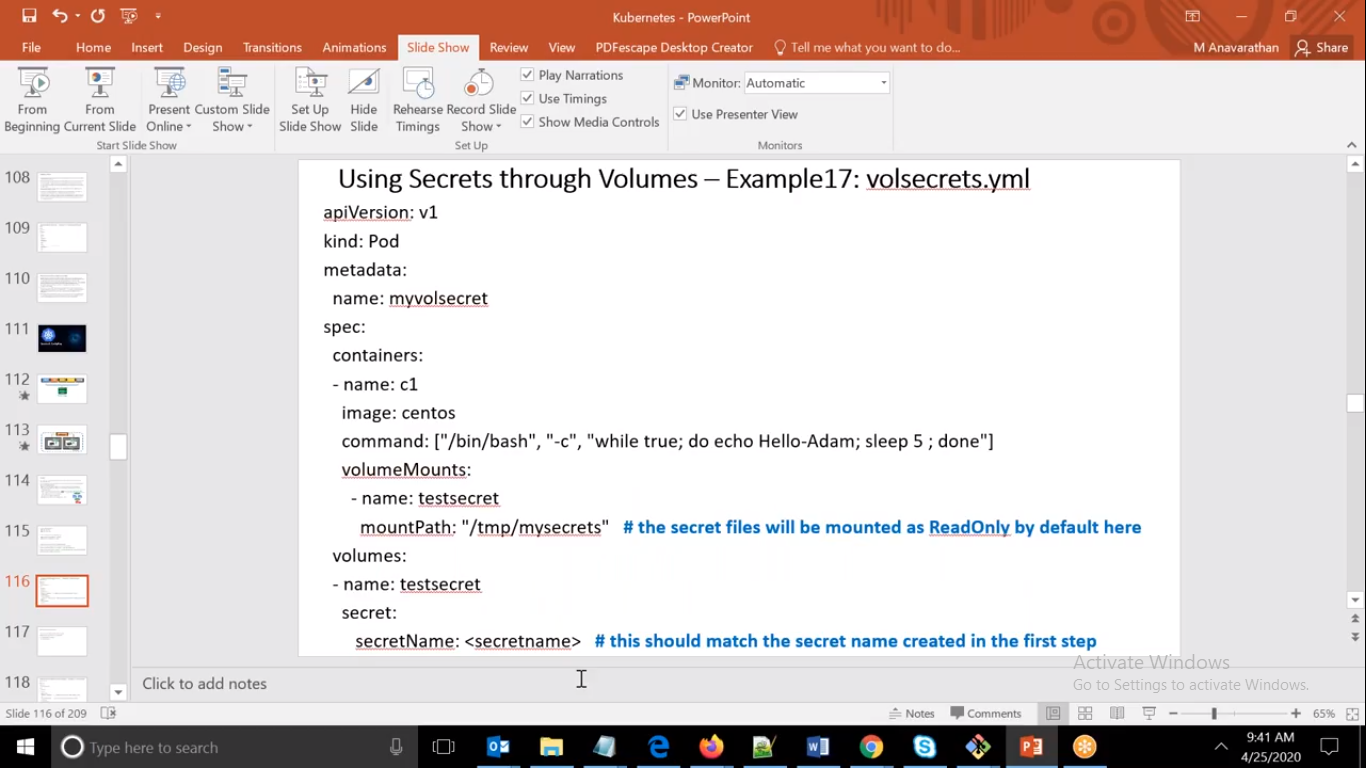


But file size should not be more than 1 MB.

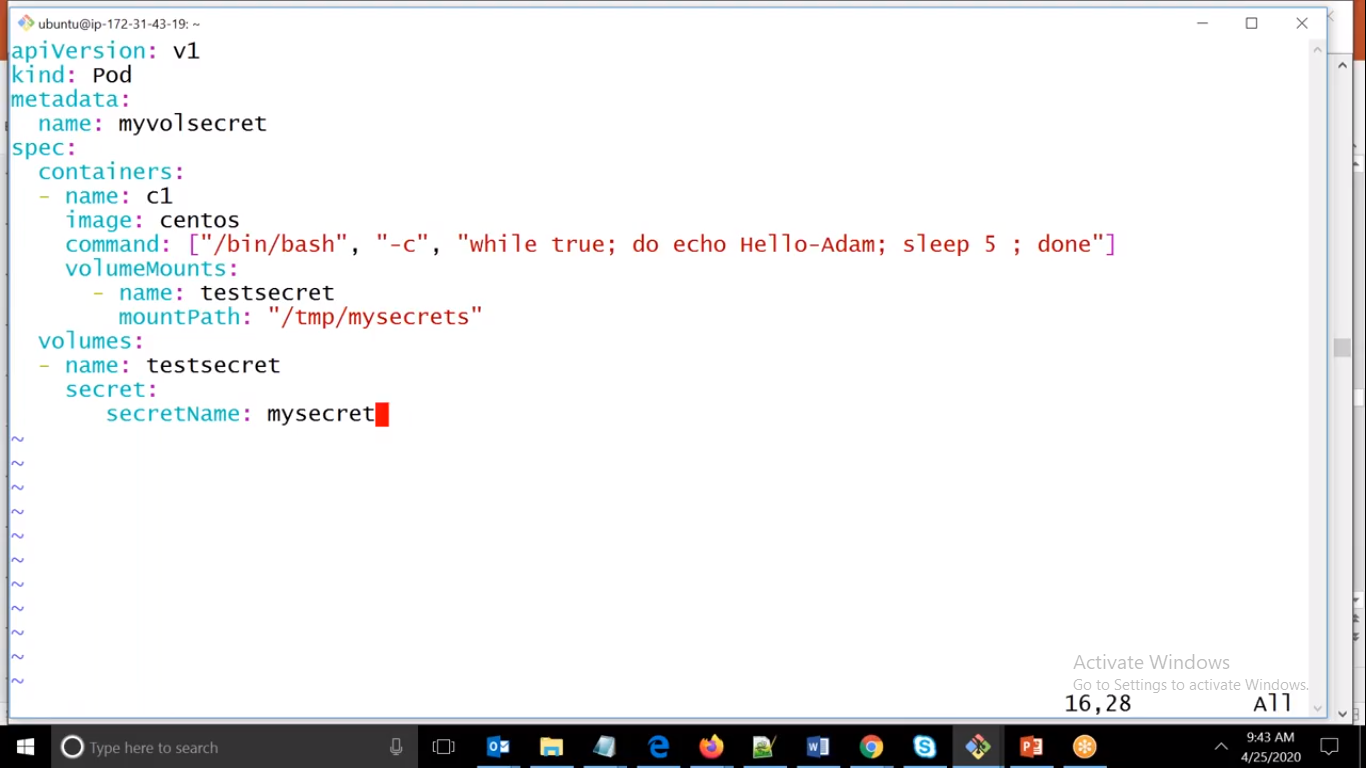
It can be text file

It can be Yaml file.





ubuntu@ip-172-31-41-23:~$ vi secret.yml



Here we have created Pod and if we see we have VolumeMounts it means that file should be placed in this specific folder while image is creating.

and if we see that we have secretName that means we have to mapped to mapped to that secret and it will share the secret files.

ubuntu@ip-172-31-41-23:~$ echo "root" > username.txt; echo "password" >password.txt

ubuntu@ip-172-31-41-23:~$ kubectl create secret generic mysecret --from-file username.txt --from-file password.txt

secret/mysecret created

ubuntu@ip-172-31-41-23:~$

Here we have create two config files and we have created secret using

kubectl create secret generic mysecret –from-file username.txt –from-file password.txt

ubuntu@ip-172-31-41-23:~$ kubectl apply -f secret.yml

pod/myvolsecret created

ubuntu@ip-172-31-41-23:~$ kubectl exec myvolsecret -it -- bash

[root@myvolsecret /]# ls

bin dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var

[root@myvolsecret /]# cd tmp/

[root@myvolsecret tmp]# ls

ks-script-\_srt3u3c ks-script-gpqu\_kuo mysecrets

[root@myvolsecret tmp]# cd mysecrets/

[root@myvolsecret mysecrets]# ls

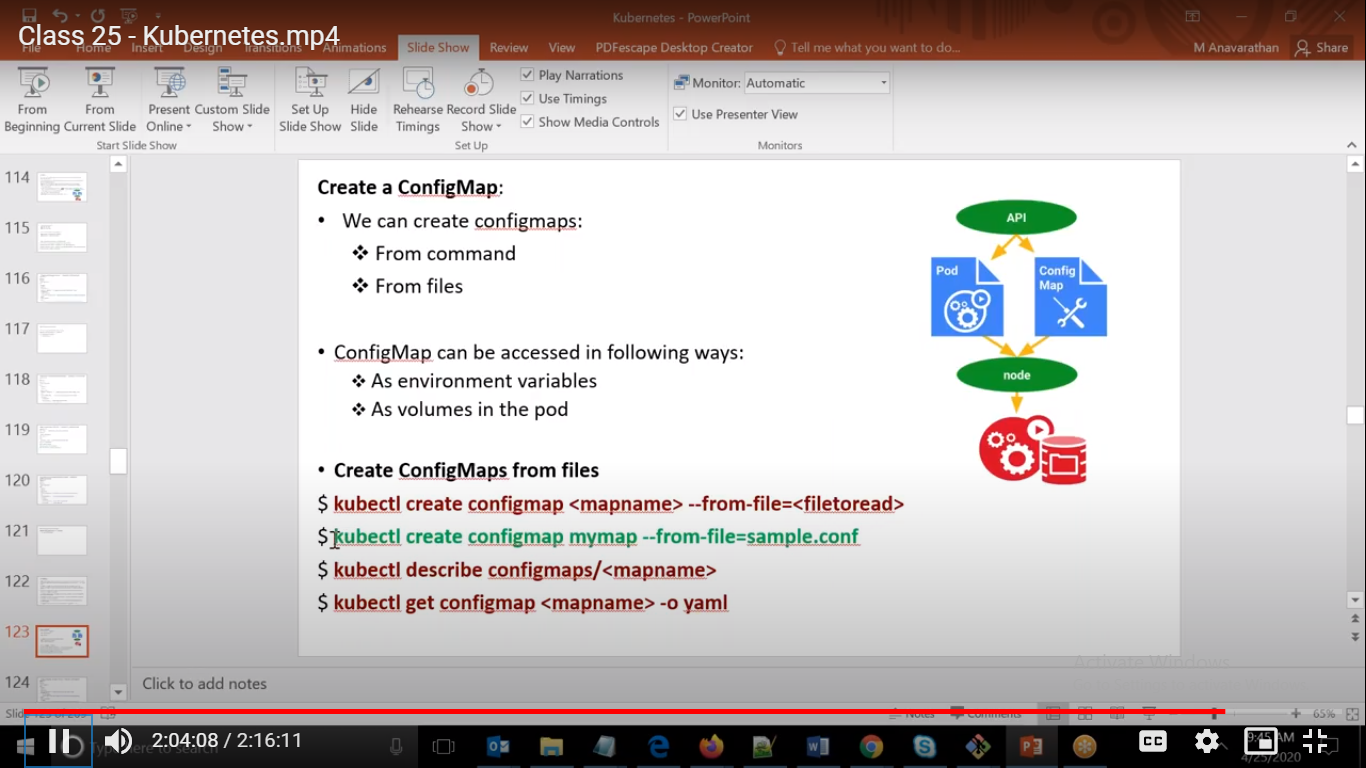
password.txt username.txt

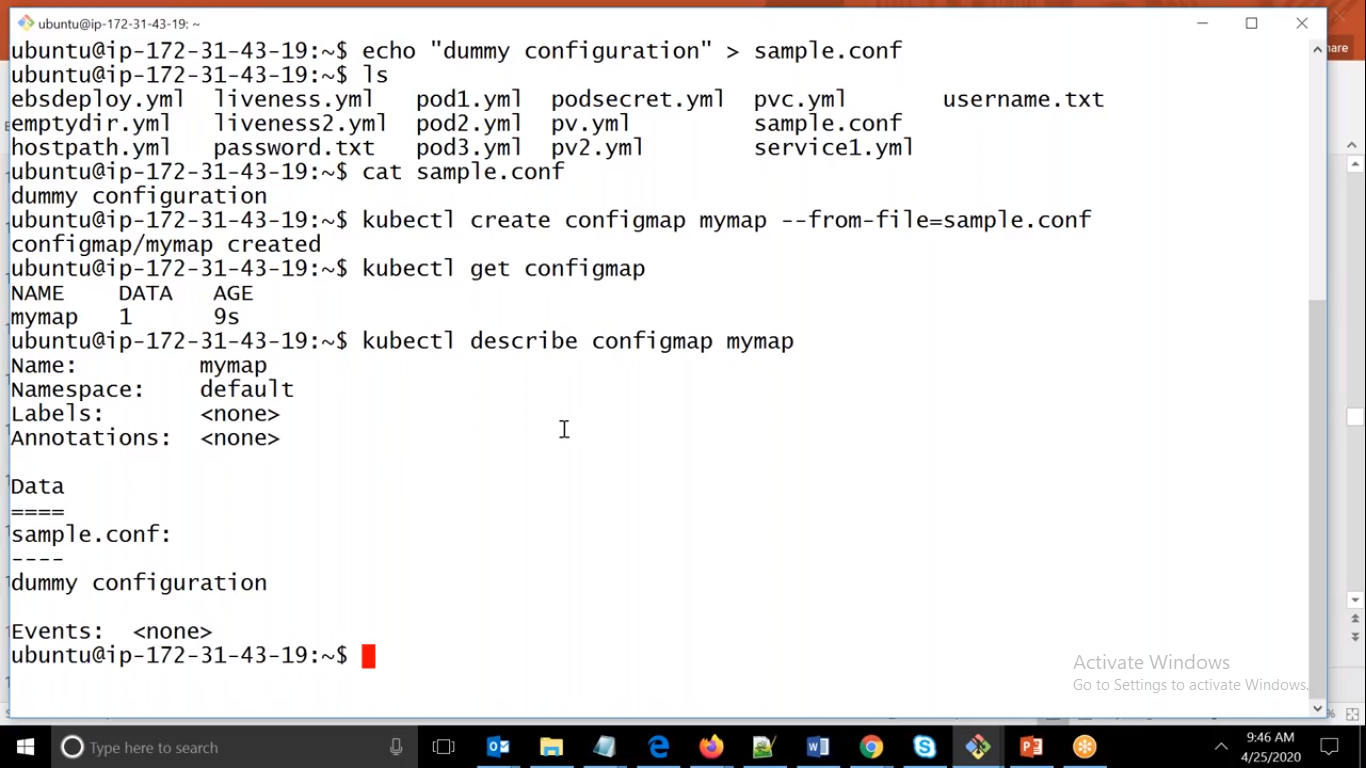
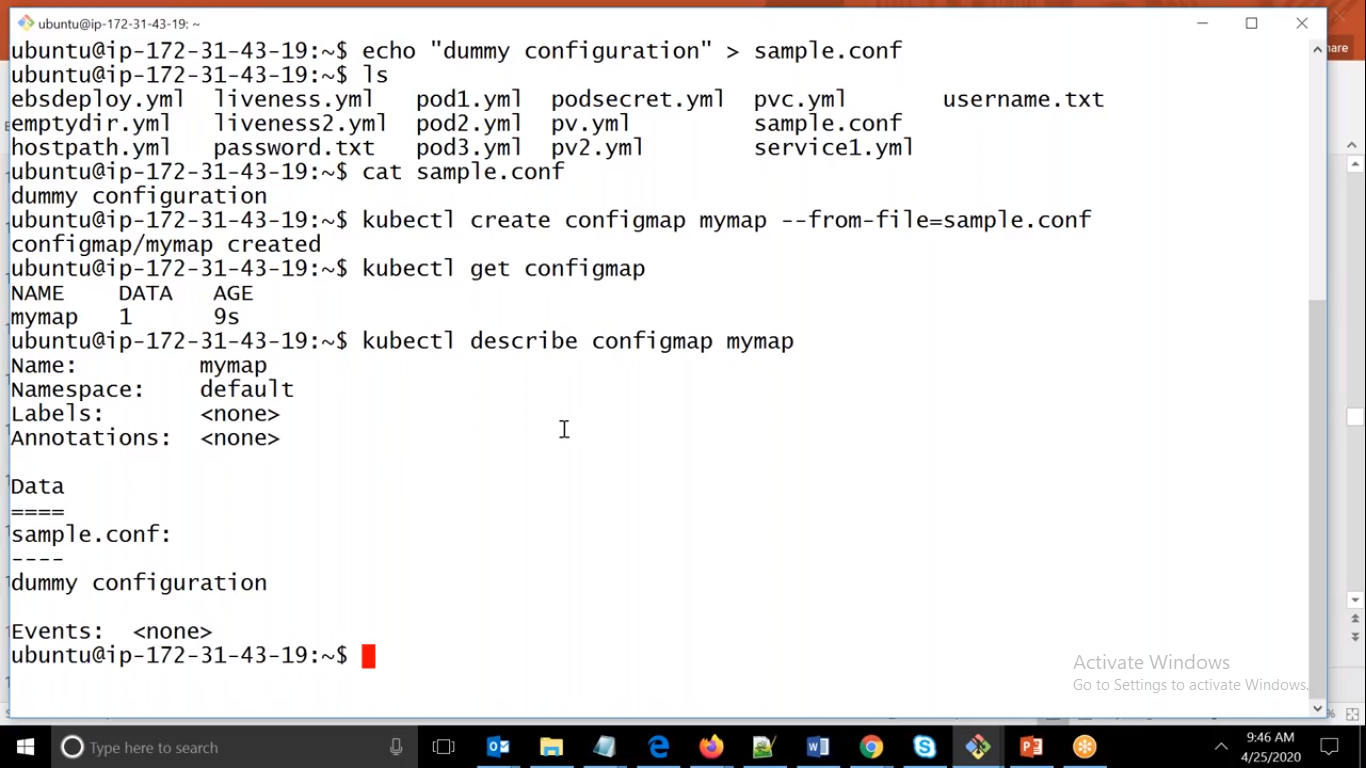
[root@myvolsecret mysecrets]# cat password.txt

password

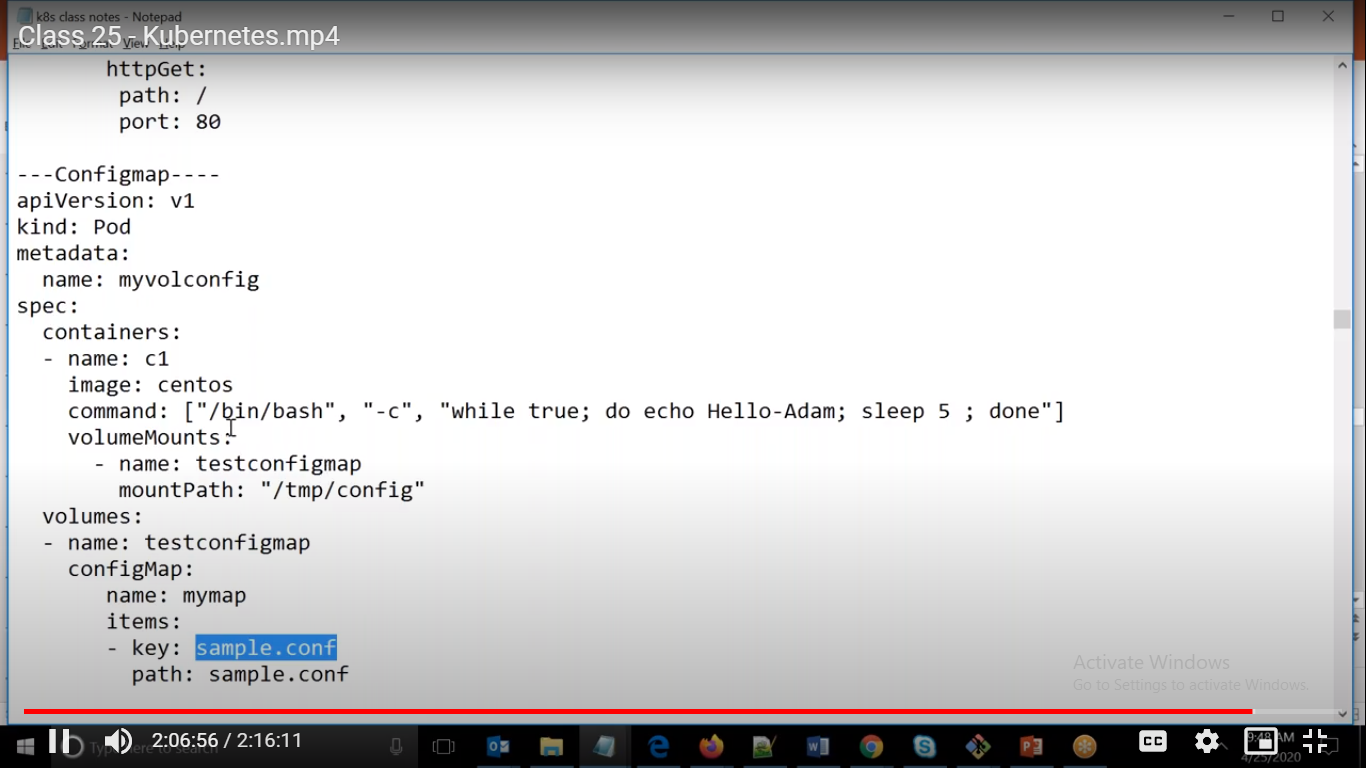
[root@myvolsecret mysecrets]#

## ConfigMap



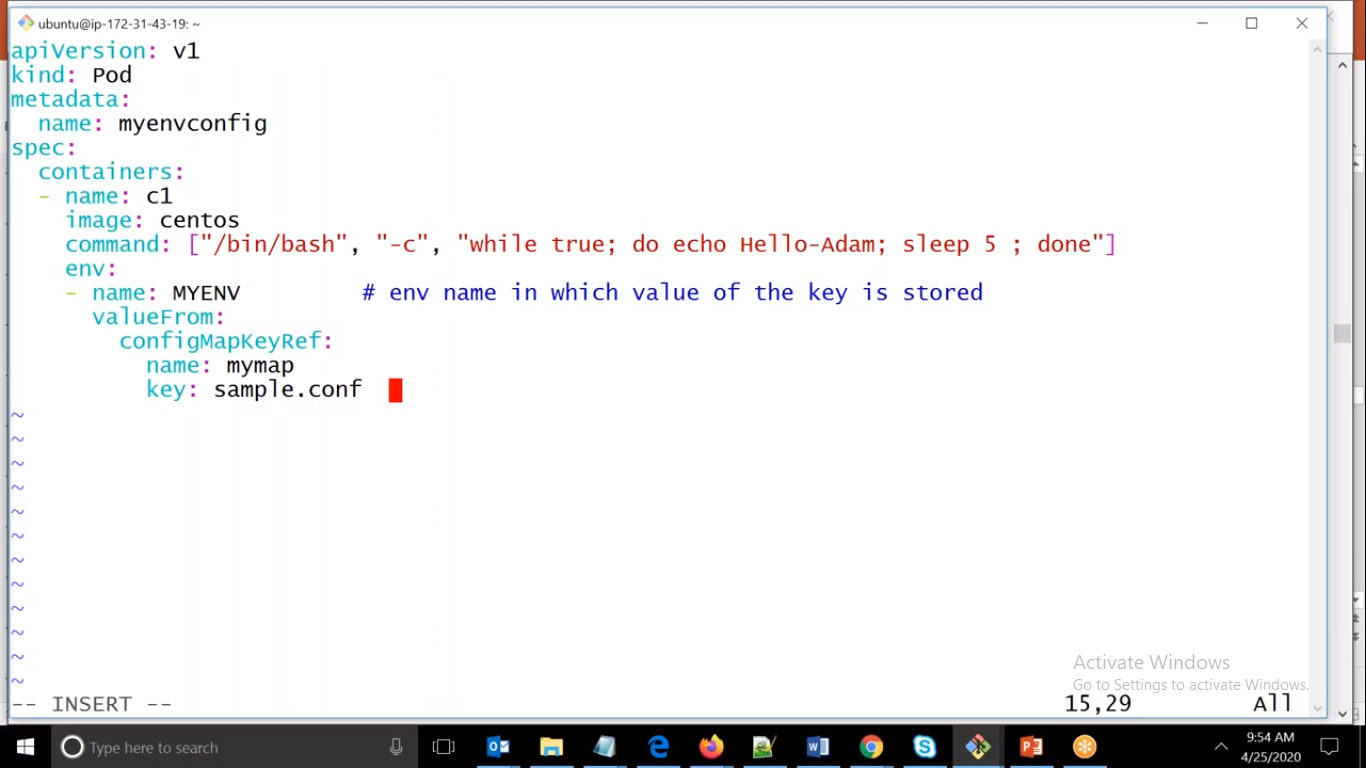
 

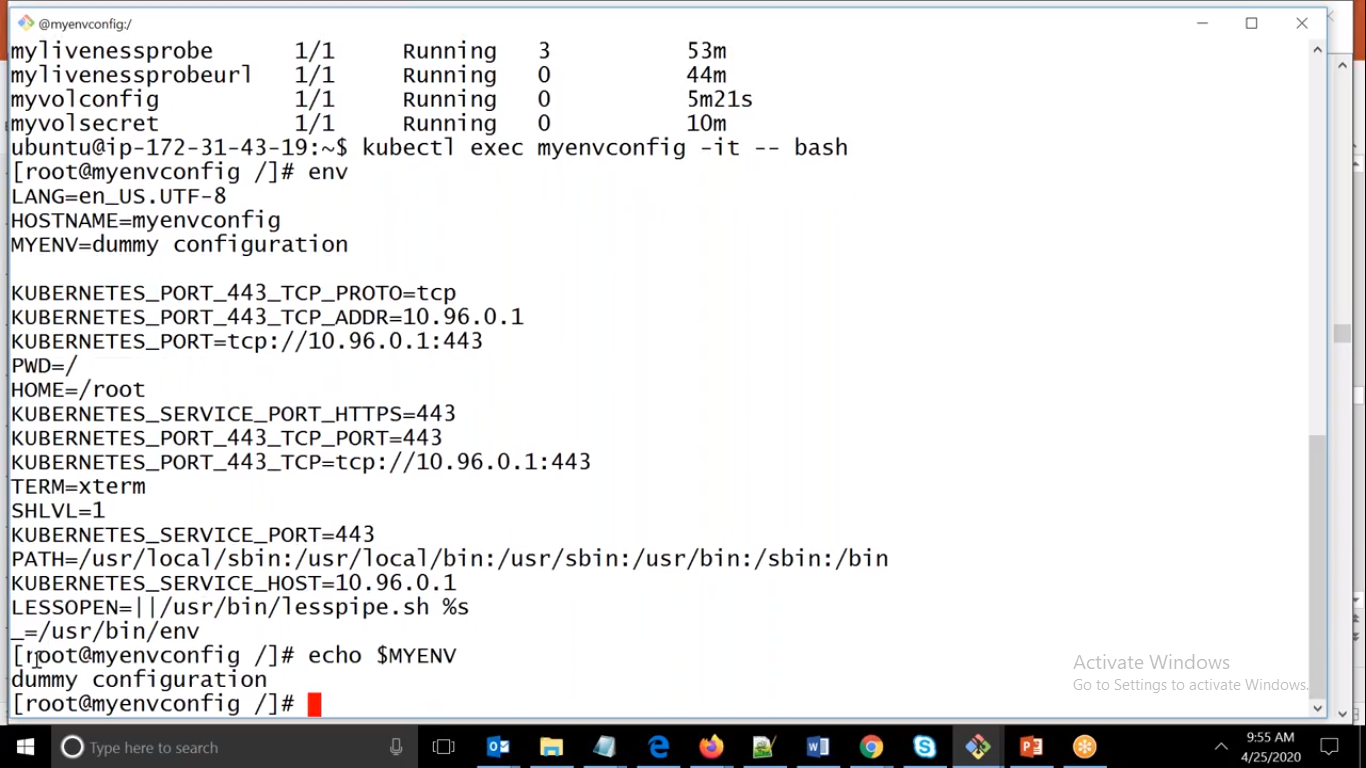
Here we have created one conf file and created configMap and if we describe it we can see that we are able to the content as well.



By this we can create image without web.conf file. and based on environment we will pass this configuration file.

We can pass the Environment variable as well and we can read that file from ConfigMap file.





Here we can see that we have created pod and via Env variable we can read it from conf file.

If you have small content then we can use for passing the Env Variable.