Lesson13 – Running Container with non-root User

In this lesson we will deploy MSSQL with persistent volume for high availability and non-root user.

The problem of running container with root user

1. Using shared volume mounted into several containers like central logging folder. The contained process running as root will have full access to every path on that mount volume. Now the application has a directory traversal vulnerability and hacker can poke around the container file system and find some config folder for "another application" on that directory. Even worse, if we add an hostpath volume, bind mount from container to the node VM like our MSMSQL container - now root process can read and write to any path on the host file system anywhere under that mount point.

2. hacker can setup a vulnerability with remote code execution in the container. When root user run the container it has a free access in the container file system to add and modify executable files, install packages and pretty much have a way in there. This raises a risk that some hacker will setup a vulnerability with remote code execution in the container.

Verify that root user run the MSSQL container:

kubectl create namespace employee

kubectl create secret generic mssql-secret --namespace=employee --from-literal='ConnectionString="server=mssql-service;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=MyDemoPwd2021!;MultipleActiveResultSets=true"' --from-literal='SA_PASSWORD=MyDemoPwd2021!'

cd C:\kuberneties\kuberneties-security\deployment

kubectl apply -f .\mssql-deploy-with-secret-and-pv.yml

kubectl get all -n employee

```
PS C:\Kuberneties\EmployeesManagement\manifests> kubectl get all -n employee
NAME                                     READY   STATUS    RESTARTS   AGE
pod/mssql-deployment-6bcb97764c-2tpl4    1/1     Running   0          4m18s

NAME                    TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
service/mssql-service   LoadBalancer   10.102.148.234   localhost     1433:30123/TCP   4m18s

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mssql-deployment    1/1     1            1           4m18s

NAME                                           DESIRED   CURRENT   READY   AGE
replicaset.apps/mssql-deployment-6bcb97764c    1         1         1       4m18s
PS C:\Kuberneties\EmployeesManagement\manifests>
```

kubectl -n employee exec -it pod/mssql-deployment-6bcb97764c-2tpl4 -- /bin/sh

whoami

exit

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   1: kubectl

PS C:\kuberneties\Kubernetes-Docker-Desktop\deployment> kubectl -n employee exec -it pod/mssql-deployment-6bcb97764c-c898d -- /bin/sh
# whoami
root
#
```

We can see that **root user** runs the MSSQL container.

Solution:

Create **mssql-deploy-with-secret-pv-non-root.yml** and add userid 10001 in the container layer as below:

```yaml
      labels:
        app: mssql
    spec:
      securityContext:
        runAsUser: 10001
        runAsGroup: 10001
      terminationGracePeriodSeconds: 10
      containers:
      - name: mssql
        image: 'mcr.microsoft.com/mssql/server:2019-CU15-ubuntu-20.04'
        resources:
          limits:
            memory: 4Gi
```

kubectl delete ns employee

kubectl create namespace employee

kubectl create secret generic mssql-secret --namespace=employee --from-literal='ConnectionString="server=mssql-service;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=MyDemoPwd2021!;MultipleActiveResultSets=true"' --from-literal='SA_PASSWORD=MyDemoPwd2021!'

kubectl apply -f .\mssql-deploy-with-secret-pv-non-root.yml

kubectl get all -n employee

```
PS C:\kuberneties\kuberneties-security\deployment> kubectl get all -n employee
NAME                                    READY   STATUS    RESTARTS   AGE
pod/mssql-deployment-5bc598bfd8-26pw7   1/1     Running   0          15m

NAME                     TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
service/mssql-service    LoadBalancer   10.99.164.54   localhost     1433:31664/TCP   15m

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mssql-deployment    1/1     1            1           15m

NAME                                           DESIRED   CURRENT   READY   AGE
replicaset.apps/mssql-deployment-5bc598bfd8    1         1         1       15m
PS C:\kuberneties\kuberneties-security\deployment>
```

kubectl -n employee exec -it pod/mssql-deployment-5bc598bfd8-26pw7 -- /bin/sh

whoami

id

ps aux

```
PS C:\kuberneties\kuberneties-security\deployment> kubectl -n employee exec -it pod/mssql-deployment-54bd89c765-dqjkb -- /bin/sh
$ whoami
mssql
$ id
uid=10001(mssql) gid=10001 groups=10001
$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mssql        1  0.1  0.3  61624 23608 ?        Ssl  16:52   0:00 /opt/mssql/bin/sqlservr
mssql       10 10.5 13.0 16071644 833188 ?     Sl   16:52   0:33 /opt/mssql/bin/sqlservr
mssql      265  0.0  0.0   2612   604 pts/0    Ss   16:57   0:00 /bin/sh
mssql      296  0.0  0.0   5900  2836 pts/0    R+   16:57   0:00 ps aux
$
```

top

q

```
top - 16:58:08 up  2:46,  0 users,  load average: 0.43, 0.64, 0.54
Tasks:   4 total,   1 running,   3 sleeping,   0 stopped,   0 zombie
%Cpu(s):  2.5 us,  2.1 sy,  0.0 ni, 94.8 id,  0.0 wa,  0.0 hi,  0.6 si,  0.0 st
MiB Mem :   6237.4 total,    128.4 free,   2158.4 used,   3950.6 buff/cache
MiB Swap:   2048.0 total,   2045.2 free,      2.8 used.   3422.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   10 mssql     20   0   15.3g 848972  61784 S   7.7  13.3   0:36.50 sqlservr
    1 mssql     20   0   61624  23608   9992 S   0.0   0.4   0:00.46 sqlservr
  265 mssql     20   0    2612    604    540 S   0.0   0.0   0:00.01 sh
  313 mssql     20   0    6144   3316   2800 R   0.0   0.1   0:00.00 top
```

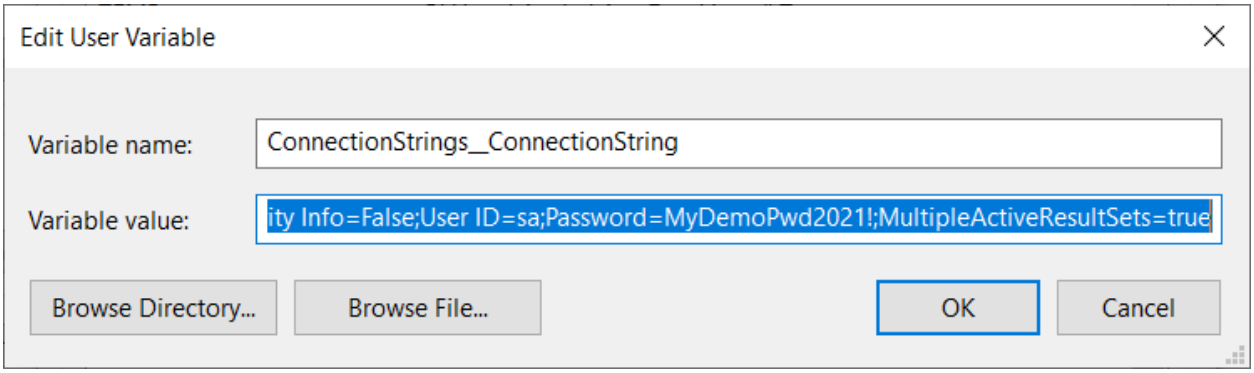
We can see that **non-root user** running the MSSQL container process with **user mssql (10001)**

cd C:\kuberneties\kuberneties-security\Employees

dotnet ef database update

NOTE: before running the command above make sure that the connection string exists in User Variable and close VScode and CMD and run again:

server=localhost,1433;Initial Catalog=EmployeeDB;Persist Security Info=False;User ID=sa;Password=MyDemoPwd2021!;MultipleActiveResultSets=true



Done!