Demo: Docker Socket Mount:DIND

Docker-in-Docker (DIND) is a concept where you run a Docker daemon inside a Docker container. The UNIX socket /var/run/docker.sock is used by the Docker daemon to interact with the Docker client. If a Docker container has access to this socket (through a volume mount), it essentially has control over the Docker daemon, which runs with root privileges on the host.

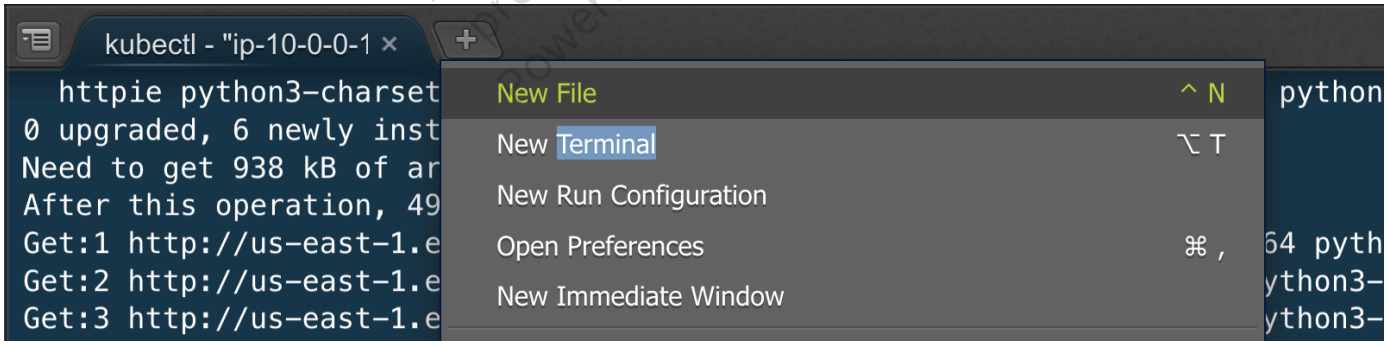
If the process running inside the container can execute arbitrary commands (as in a command injection vulnerability), an attacker can abuse this to interact with the Docker daemon through the socket.

Trainer-Specific Demo Setup (Not for Students)

Open New Terminal (Optional)

If current working directory is not `workspace/course` .

- Click on `+` icon, then select `new terminal` to open new terminal.



- Keep current working directory as `workspace/course`

```
cd course/4.6_misconfigkind_scenario/dind
ls
```

```
root@ip-10-0-0-153:/home/ubuntu/ workspace# cd course/4.6_misconfigkind_scenario/dind
root@ip-10-0-0-153:/home/ubuntu/ workspace/course/4.6_misconfigkind_scenario/dind# ls
custom-dind-deployment.yaml
```

```
kubectl apply -f custom-dind-deployment.yaml
```

```
root@ip-10-0-0-153:/home/ubuntu/ workspace/course/4.6_misconfigkind_scenario/dind# kubectl apply -f custom-dind-deployment.yaml
deployment.apps/custom-dind-deployment created
```

- Get a shell inside the custom-dind-container

Wait for 10 seconds in case of error.

```
kubectl exec -it deploy/custom-dind-deployment -- sh -c "docker images"
```

```
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

- Now we can access the host system by running the following docker commands with passing docker.sock UNIX socket

```
kubectl exec -it deploy/custom-dind-deployment -- sh -c "docker -H
unix:///custom/docker/docker.sock images"
```

```
root@ip-10-0-0-153:/home/ubuntu/ workspace/course/4.6_misconfigkind_scenario/dind# kubectl exec -it deploy/custom-dind-deployment -- sh -c "docker -H unix:///custom/docker/docker.sock images"
REPOSITORY     TAG       IMAGE ID       CREATED        SIZE
kindest/node   <none>    d8644f660df0   6 months ago   898MB
root@ip-10-0-0-153:/home/ubuntu/ workspace/course/4.6_misconfigkind_scenario/dind#
```


- Explanation:
  - When running the `docker images` command directly inside the container without explicitly specifying the Docker socket to use. By default, the Docker client inside the container tries to connect to the default Docker socket, which is /var/run /docker.sock. However, a custom Docker socket is mounted from the host to the container at /custom/docker/docker.sock, the default socket is not accessible.
  - While using `docker -H unix:///custom/docker/docker.sock images` , try to explicitly specify the Docker socket to use with the -H flag. By providing the path to the custom Docker socket, unix:///custom/docker/docker.sock, the Docker client inside the container connects to that socket for executing the command.
  - As a result, the first command shows a blank output because the Docker client inside the container is not able to connect to the default Docker socket. The second command successfully connects to the custom Docker socket and shows the kindest/node image.

Cleanup

- Run the `kubectl delete` command to remove the pods running.

```
kubectl delete -f custom-dind-deployment.yaml
```

Wait for the pods to be deleted.
