class-vmware-docker-2018-03-12

vmware advanced docker class page

View on GitHub

docker namespace exploration

Explore pid namespace

Take a look at the current processes

```
ps aux
```

Create a new namespace and run bash:

```
sudo unshare --fork --pid --mount-proc bash
```

Take a look around:

```
ps aux
```

What is pid 1 in the new namespace?

Inside the new namespace, run a long lived process like `top`. Then in a separate ssh connection to the docker host look at the current processes

```
ps aux
```

Find the namespaced processes? What PIDs do they have?

Feel free to explore with other commands. When you're done exploring just `exit`

Explore docker namespaces

Run any container, for example:

```
#sleep for 5 minutes and exit
docker run --detach alpine sleep 300
```

Copy the container id for your running container, this is $CID below

If you don't already have the CID, you can find it with `docker ps`

docker exec

Use the `docker exec` utility to explore the inside of the container:

```
docker exec -it $CID /bin/sh
#Explore within the container, e.g.
ps aux
ls /
# ...
# when you are done just exit
exit
```

Now use the more general linux `nsenter` utility to enter the container's namespaces

```
PID=$(docker inspect --format  $CID)
sudo nsenter --target $PID --mount --uts --ipc --net --pid /bin/sh
```

Explore the memory cgroup limits

```
sudo cgcreate -a play -g memory:playground
```

See what's in it

```
ls -la /sys/fs/cgroup/memory/playground/
```

`memory.limit_in_bytes` looks useful, let's try tweaking that.

Let's set a 10MiB memory limit for the cgroup

```
echo 10485760 | tee /sys/fs/cgroup/memory/playground/memory.limit_in_bytes
```

Let's try running a memory hogging program:

```
#Will use 100MiB
memhog 104857600 1
```

Now let's join the cgroup and see what happens

```
sudo cgexec -g memory:playground bash

#Will use 10MiB
memhog 10485760 1
```

You should see this:

> allocating 10485760 bytes... writing 10485760 bytes... Killed

What is the exit code? What does this exit code signify?

```
echo $?
```

If you want to delete the cgroup you can run

```
sudo cgdelete memory:playground
```

credits: Much of this example taken from https://jvns.ca/blog/2016/10/10/what-even-is-a-container/
