# DevOps Tools

## Day - 17

**Prakash Ramamurthy**
prakash.ramamurthy@wipro.com

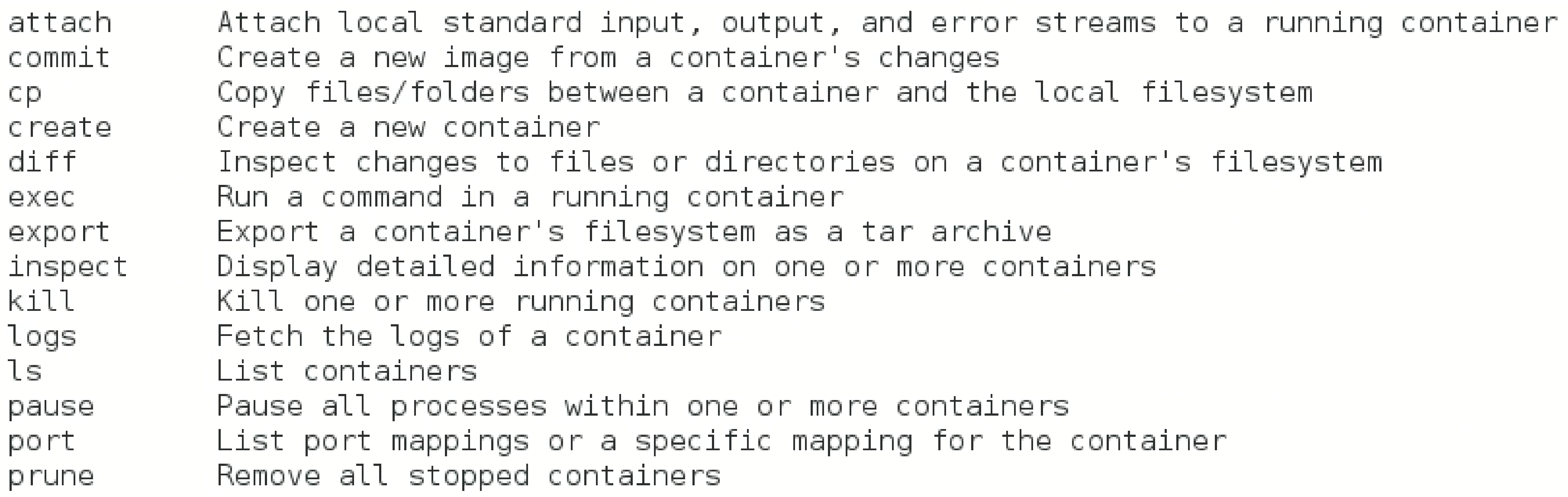
# Agenda

- Docker Containers & Images
- Docker Networking
- Docker Volumes
- Hands-On Demonstration

# Docker Containers and Images

# Docker Containers

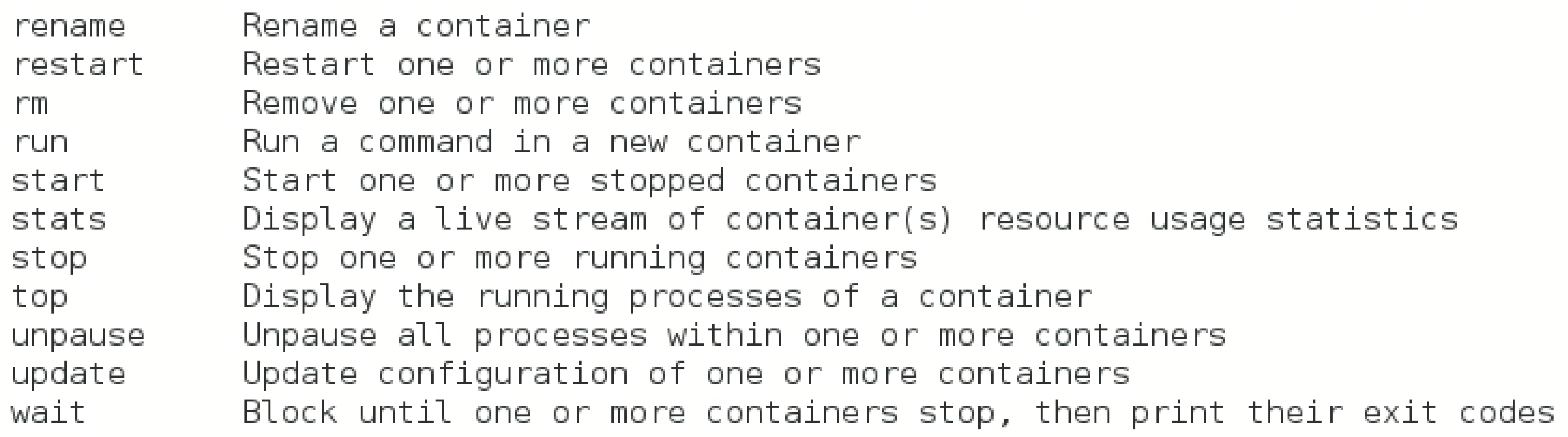
## Docker Container Operations



```
attach      Attach local standard input, output, and error streams to a running container
commit      Create a new image from a container's changes
cp          Copy files/folders between a container and the local filesystem
create      Create a new container
diff        Inspect changes to files or directories on a container's filesystem
exec        Run a command in a running container
export      Export a container's filesystem as a tar archive
inspect     Display detailed information on one or more containers
kill        Kill one or more running containers
logs        Fetch the logs of a container
ls          List containers
pause       Pause all processes within one or more containers
port        List port mappings or a specific mapping for the container
prune       Remove all stopped containers
```

# Docker Containers

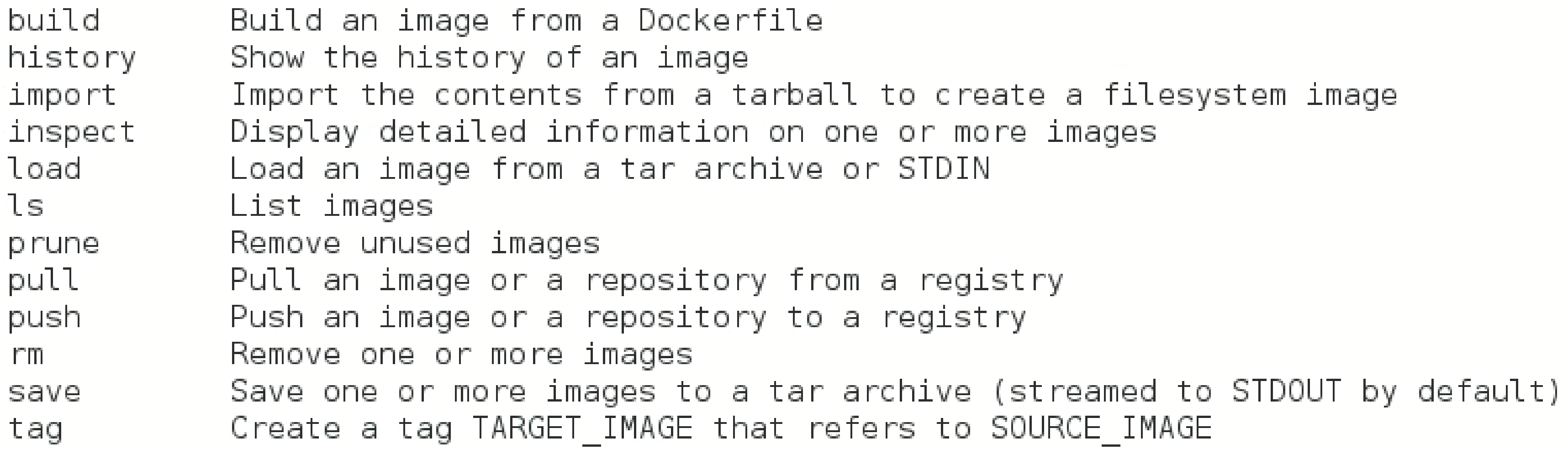
## Docker Container Operations



```
rename      Rename a container
restart     Restart one or more containers
rm          Remove one or more containers
run         Run a command in a new container
start       Start one or more stopped containers
stats       Display a live stream of container(s) resource usage statistics
stop        Stop one or more running containers
top         Display the running processes of a container
unpause     Unpause all processes within one or more containers
update      Update configuration of one or more containers
wait        Block until one or more containers stop, then print their exit codes
```

# Docker Images

## Docker Image Operations

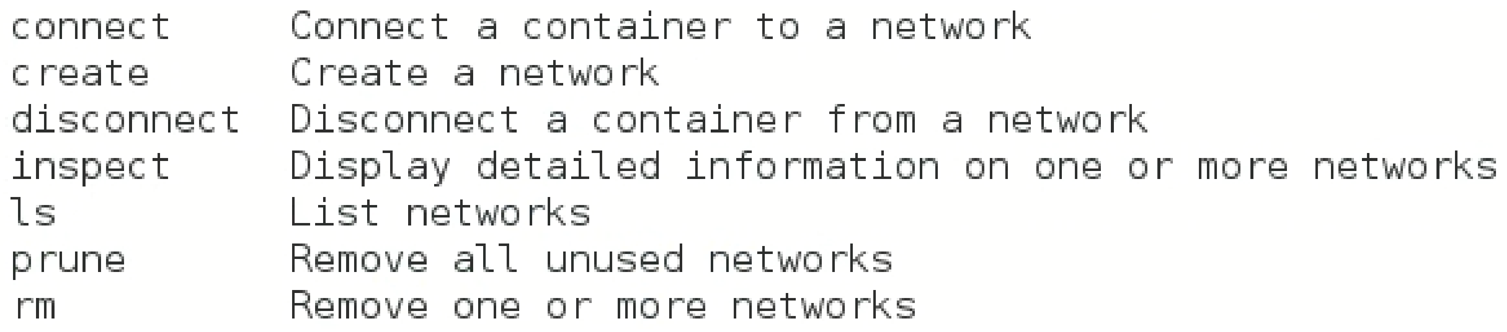


```
build       Build an image from a Dockerfile
history     Show the history of an image
import      Import the contents from a tarball to create a filesystem image
inspect     Display detailed information on one or more images
load        Load an image from a tar archive or STDIN
ls          List images
prune       Remove unused images
pull        Pull an image or a repository from a registry
push        Push an image or a repository to a registry
rm          Remove one or more images
save        Save one or more images to a tar archive (streamed to STDOUT by default)
tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
```

## Docker Networking

## Docker Networking

### Network communication between containers

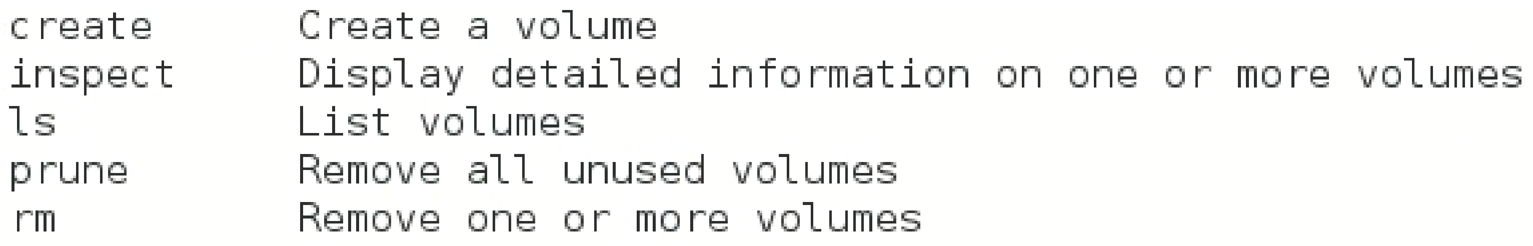


```
connect      Connect a container to a network
create       Create a network
disconnect   Disconnect a container from a network
inspect      Display detailed information on one or more networks
ls           List networks
prune        Remove all unused networks
rm           Remove one or more networks
```

## Docker Volumes

# Docker Volumes

## Sharing data through volumes



```
create      Create a volume
inspect     Display detailed information on one or more volumes
ls          List volumes
prune       Remove all unused volumes
rm          Remove one or more volumes
```

# Hands-On Demonstration

## Handson Demonstration

### Start, Stop, Restart and Kill

- Start the container

```
$ docker container start app_doc
```

- Stop the container

```
$ docker container stop app_doc
```

- Kill the container

```
$ docker container kill app_doc
```

- Restart the container

```
$ docker container restart app_doc
```

- Note that the above operations actually do same thing to containerized application

## Handson Demonstration

### Pause the container

- Pause the container

```
$ docker container pause app_doc
```

- Unpause the container

```
$ docker container unpause app_doc
```

- Actually you are passing the containerized application
- Check whether the paused container responds to commands or any external triggers
- What is the difference between stopping and pausing of containers

## Handson Demonstration

### Wait for an application to end

- Command to trigger wait operation on a container

```
$ docker container wait app_doc
```

- You are actually waiting for an application (container) to end.
- Exit from container with exitcode
- What is the use of exitcode for application exits

# Handson Demonstration

## Execute a command remotely inside a container

- To execute a linux command remotely inside container

```
$ docker container exec app_doc ls /usr
$ docker container exec app_doc pwd
```

- Check the file content inside the container
- How is this different from interacting with the container

# Handson Demonstration

## Check on container activity on file system

- Container log for shell command activity

```
$ docker container logs app_doc
```

- Do some file specific operations on a container
- Check whether those commands and their outputs get listed with logs

- Look for change happened to container file system

```
$ docker container diff app_doc
```

- Add some folders and files
- Make some changes to files
- Check whether diff can list all such changes made to file system

## Handson Demonstration

### Tracing the Process tree

- Check the process id of running container

```
$ docker container top app_doc
```

- Looking up the process tree

```
$ ps -ef | grep 12892
$ ps -ef | grep 12877
$ ps -ef | grep 12261
$ ps -ef | grep 12229
$ ps -ef | grep 1
```

- Each container and virtual machine is basically a process on the system
- Track the container process and its parents in the system till up to init process

## Handson Demonstration

### Tracking system events across containers

- Track the system events

```
$ docker system events
```

- With container events, helps you track the status of application.
- Helps the system to know when which application is started, stopped, restarted
- Also helps to track how many application (containerized) applications are running in the system

- Initiate multiple system events to track them

```
$ docker container start app_doc
$ docker container attach app_doc
:/# exit
$ docker container restart app_doc
```

# Handson Demonstration

## Tracking system resource utilization by container

- Command to track the resource utilization by a container

```
$ docker container stats app_doc
```

- Make changes to container to track change in resource use interactively
- By creating a file
- By creating a folder
- By making changes to file
- Any other file system specific events.

## Handson Demonstration

### Launching WebApp in tomcat container

- Launch tomcat container

```
$ docker container run -d -P tomcat:8
$ docker container run -d -p 11022:8080 --name tomcat_app tomcat:8
```

- List the port mapping for tomcat container

```
$ docker container ls
$ docker container port optimistic_bardeen
```

- Copying war file into a container

```
$ docker container cp /home/osgdev/dockerlab/NewApp1.war
optimistic_bardeen:/usr/local/tomcat/webapps
```

- Check your app on browser: `http://localhost:11022/NewApp1`

# Handson Demonstration

## Docker network

- Listing Docker Network subcommands

```
$ docker network
```

- List the network

```
$ docker network ls
```

- Details of the network

```
$ docker network inspect bridge
```

## Handson Demonstration

### Docker network

- Creating New Network

```
$ docker network create new-net
```

- Creating container connecting to new network

```
$ docker container run -it --net new-net ubuntu:14.04
```

- Deleting the network

```
$ docker network rm new-net
```

- Create a new network
- Create a container that joins the new network

# Handson Demonstration

## Docker network

- Connecting a container to specific network

```
$ docker network connect bridge
```

- Disconnecting a container from specific network

```
$ docker network disconnect new-net
```

- Connect a container on one network to another and check the network interfaces and IP
- Disconnect the container connected on two networks and check the network interfaces and IP

## Handson Demonstration

### Docker Volume sharing with system folder

- Sharing a system folder with a volume inside the container

```
$ docker container run -it -v /home/osgdev/dockerlab/SHARE/:/FROMHOST
ubuntu:16.04
```

- Extending system folder sharing from one container to another

```
$ docker container run -it --volumes-from 048e62898477 ubuntu:16.04
```

- Explore use of volumes as a means to share data between system and containers

## Handson Demonstration

### Using shared folder concept for Webapp

- Sharing system folder having war file with webapps folder

```
$ docker container run -d -p 11055:8080 -v /home/osgdev/dockerlab/web:/usr/local
```

- Launch one container with above command
- Launch another container copy the war file to webapp folder
- Compare the webapp folder in both containers for difference between folders and volumes

## Handson Demonstration

### Docker Volume

- Docker Volume subcommands

```
$ docker volume
```

- List the volumes

```
$ docker volume ls
```

- Create a volume

```
$ docker volume create foo
```

- Get Volume details

```
$ docker volume inspect foo
```

- Remove the volume

```
$ docker volume rm foo
$ docker volume prune
```

# Handson Demonstration

## Another way to mount volume and share content

- Mount volume inside containers

```
$ docker container run -it -v foo:/home/share_vol ubuntu:16.04
$ docker container run -it -v foo:/opt/vol_res ubuntu:16.04
```

- Check whether these volumes can be used to share the content between two containers
- Delete both the containers
- Launch another container mounting the same volume
- If the new container having the volume mounted has same data then your volume is persistent
- Check where this volume content is actually available on the system

## Thank You