

Distributed Systems Study Group

Second Meeting: Dec 17rd, 2019

Gerry Seidman
gerry@iamx.com

Agenda (approximately)

- Check-in: Who has something to share/present?
 - Review Tasks
- Hands-on (get VMs Set up)
- Demo of AWS and GCE VM setup (Michael/Teresa + Keith/Noah)
- More on Docker and Kubernetes (Gerry)
- Hands-on time

Goals for Hands-On Tonight

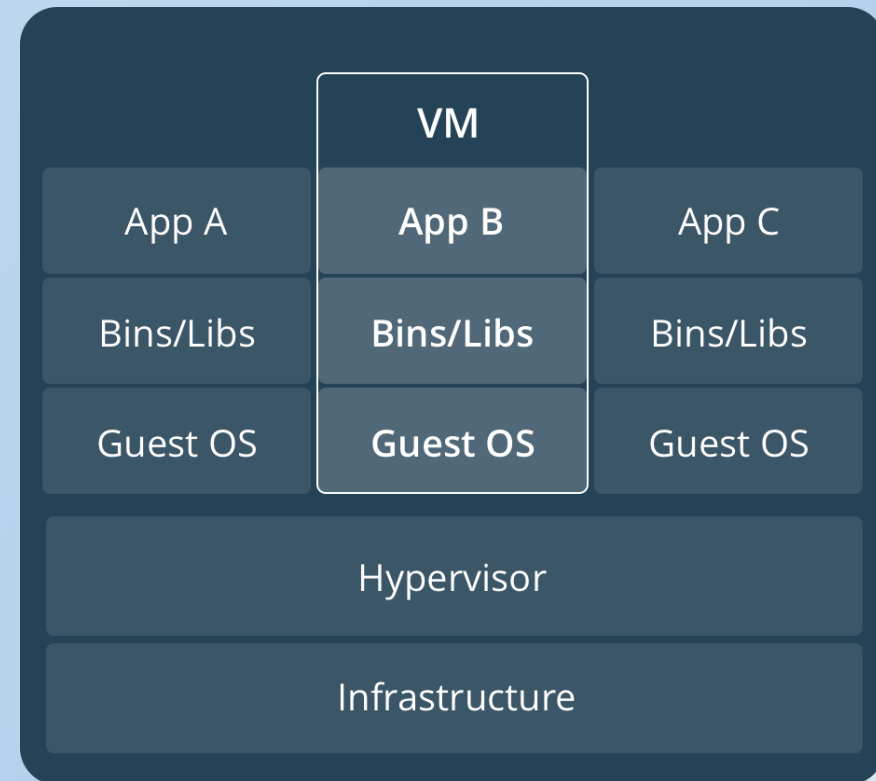
- VMs (Laptop, AWS or GCE)
 - Set up with the appropriate network inter-connectivity
 - Install Ubuntu 18.04 Linux (Headless, ie no desktop interface, only ssh)
- Have docker properly installed (including a private registry)
 - Building docker images, pushing to private registry, running docker containers
- Have a Kubernetes cluster installed
 - Configure/Deploy application in Kubernetes

Software Packaging and Deployment

- Developer writes some code, tests it locally and we're ready to deploy
 - But where?
 - How to deliver it?
- Zip it up and install it on some dedicated machine
 - Oh yeah.. There's also configuration files
- You need a machine to handle peak load of that machine
 - How much Disk? RAM? Network? GPU? Fast Processor?
 - Hard Allocation of Resources

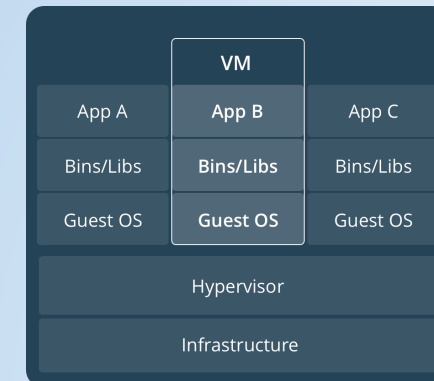
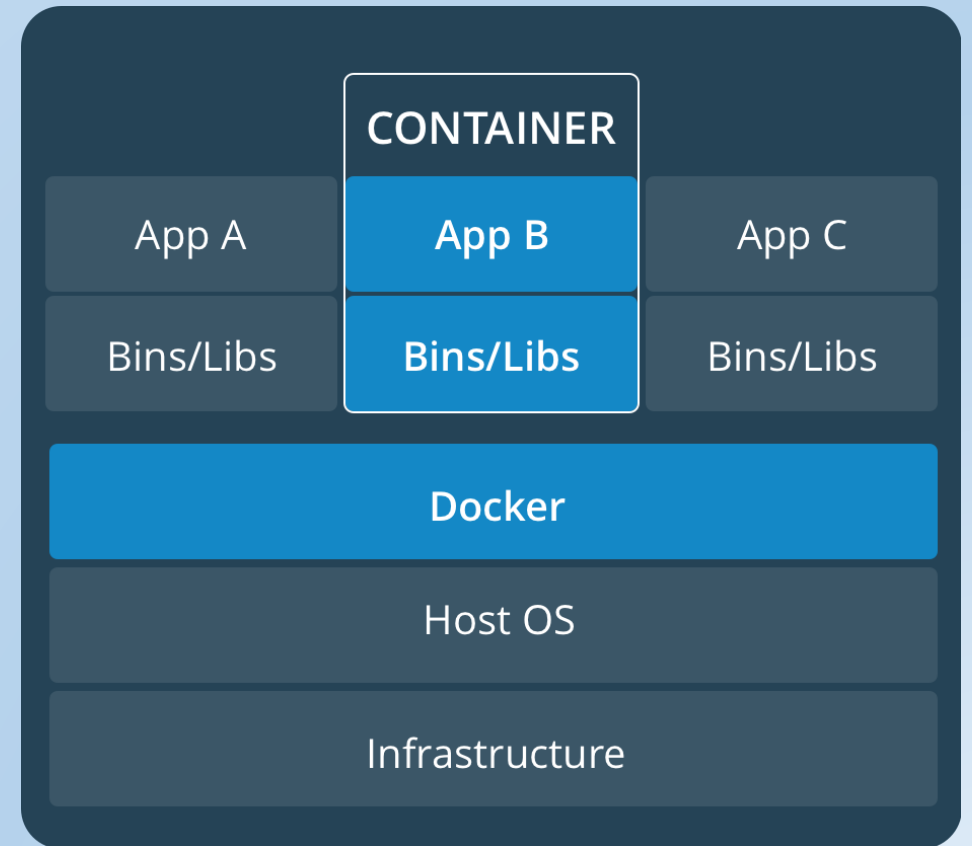
Virtual Machine

- Unit of deployment a VM Image
 - Just a big TAR file
 - When running it is a VM
- Semi-Soft Allocation of Resources
 - Disk, Memory, Network
- Slow to load
- Strong Security Boundary
- Hard to maintain
 - Same amount of work as a dedicated Machine
 - Security patches



Container

- Unit of deployment a Container Image
 - Just a big TAR file
 - When running it is a Container
- Soft Allocation of Resources
 - Disk, Memory
 - Some shared across Containers
- Fast to load
 - Run as a Host Process
- Pretty Strong Security Boundary
 - Linux Kernel
- Easier to maintain
 - Host OS

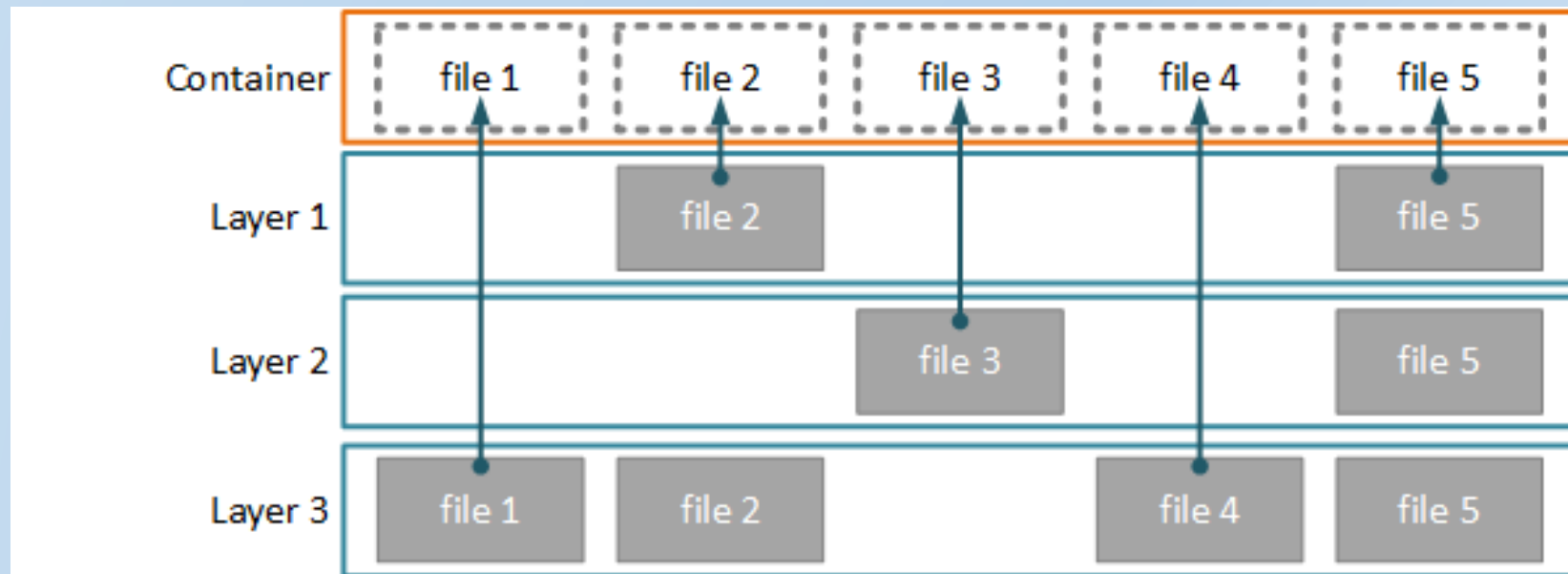


What is Docker

- Docker File Format for Container Images
- Docker Container Image builder
 - Must have Root access to build
- Docker Runtime 'server'
 - dockerd
 - REST or CLI Client
 - Must have Root access to launch containers
- Docker Registry
- Other Container implementations
 - Singularity
 - Podman/Buildah

Overlay File System – Container Image Layers

- Copy on Write File System



Container Resource Management

- Container Isolation
 - Control Groups (cgroups)
 - Namespaces

Namespaces

- The pid namespace: Process isolation (PID: Process ID).
- The net namespace: Managing network interfaces (NET: Networking).
- The ipc namespace: Managing access to IPC resources (IPC: InterProcess Communication).
- The mnt namespace: Managing filesystem mount points (MNT: Mount).
- The uts namespace: Isolating kernel and version identifiers. (UTS: Unix Timesharing System).

Docker Simplest Example

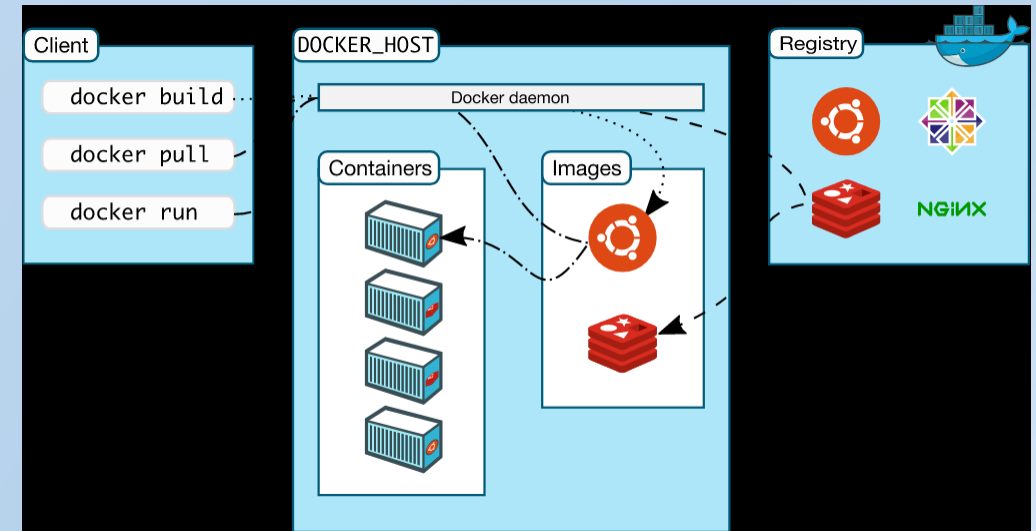
```
$ sudo docker run -i -t debian /bin/bash
```

```
$ docker run debian echo hello-world  
hello-world
```

- <http://containertutorials.com/>

Docker Notes

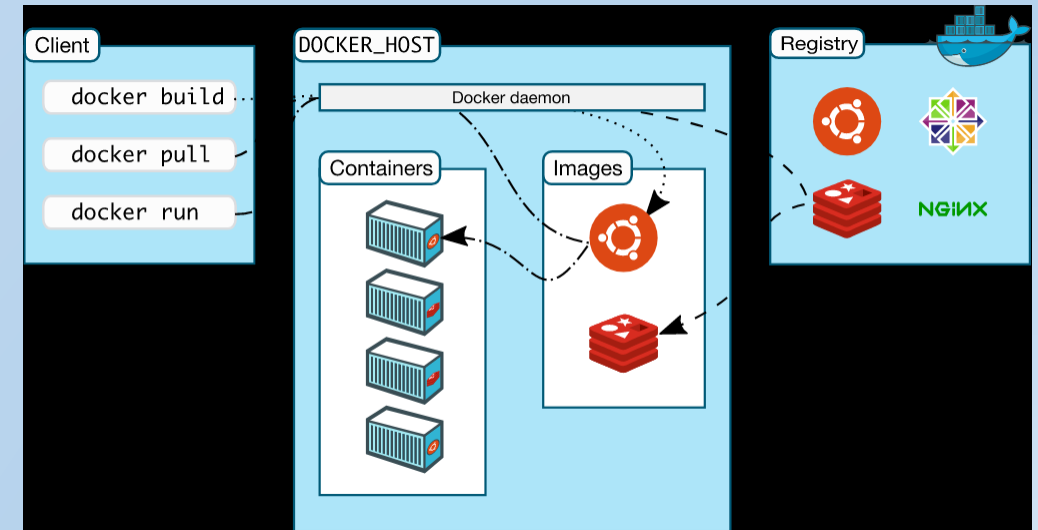
- Docker Object
 - Images
 - `docker build`
 - Containers
 - `docker run -i -t ubuntu /bin/bash`
- Will pull from configured registries
- Repositories and Registries
 - Build images into/run-from Repository
 - Push/pull images to/from Registries
- Need to do everything as root



(analogy: Git local repo)
(analogy: GitHub)

Understanding Images

- Registry vs local Repository
 - Public (dockerhub) vs Private registries
 - docker search alpine-apache
- docker images
 - Image variants [image]:[tag]
- docker build
- docker push
- docker pull



Pull the alpine image,

```
$ docker pull alpine
```

Check IP Address of the container

```
$ docker run alpine ifconfig
```

Launching a bash shell

```
$ docker run -i -t alpine /bin/bash
```

Docker Build Example

1. Create a Dockerfile

```
FROM smebberson/alpine-apache
ADD ./public-html/myindex.html /var/www/localhost/htdocs
```

2. Create a directory public_html with the following content in myindex.

```
<html>
<body>
Hi There - Static page served by Apache Server
</body>
</html>
```

3. Your directory should look like this

```
$ tree .
.
├── Dockerfile
└── public-html
    └── myindex.html
```

4. Create a Docker image

```
$ docker build -t my-apache2-alpine .
```

This will create a my-apache2 image.

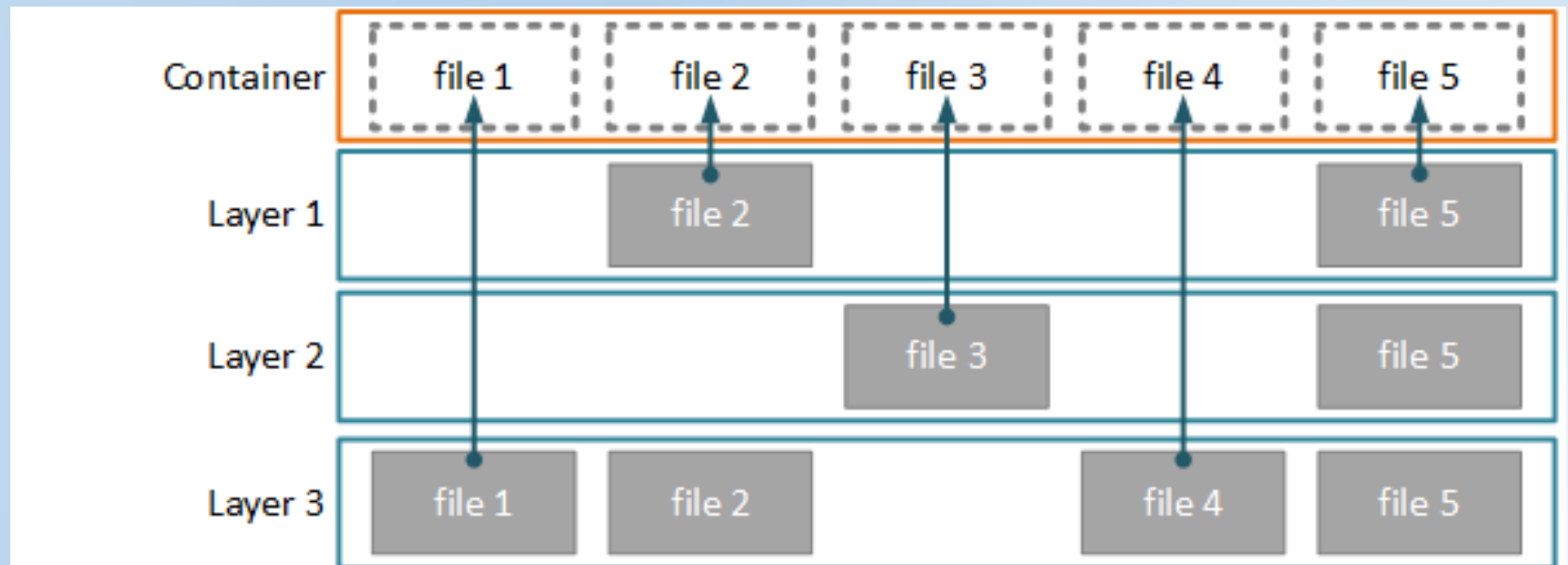
5. Create a Docker Container running this image

```
docker run -p 80:80 --name my-apache2-alpine-1 my-apache2-alpine
```

Dockerfile Commands

- FROM
- ADD
- RUN
- COPY
- EXPOSE
- ENVIRONMENT

- Each RUN creates a Layer
 - To Reduce Layers use “&”
 - RUN <cmd> & <cmd> & <cmd>



Container Tutorial - Getting Started

- [http://containertutorials.com/get started](http://containertutorials.com/get_started)
- Small Linux's
 - <http://containertutorials.com/alpine.html>
 - Busybox
- Gerry's Docker Scripts
 - <https://github.com/GerrySeidman/Distributed-Systems-Study-Group/blob/master/docker/README.md>

Container Tutorial – Docker Images

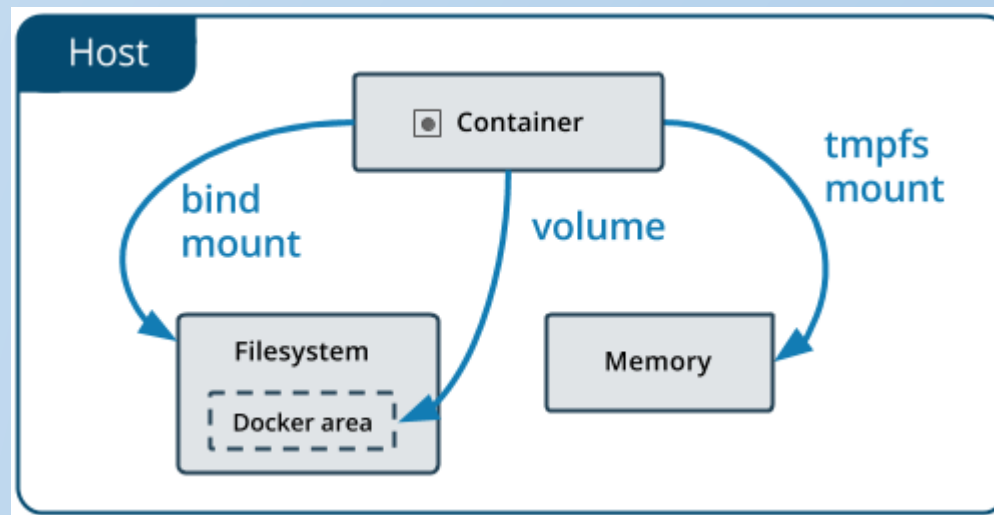
- <http://containertutorials.com/images.html>

Container Tutorial – Docker Registry

- <http://containertutorials.com/registry.html>

Volumes and Network Port Binding

- Mount Host Directory as a 'volume'
 - Ugh overloaded term.. hear 'volume' means a rooted directory tree.



- Networking
 - Bind Host port to container port

Container Tutorial – Docker Network Ports

- http://containertutorials.com/network/basic_network.html

Container Tutorial – Docker Volumes

- <http://containertutorials.com/volumes.html>

docker Misc Commands

- docker info
- docker inspect <container>

Important Topics

- Docker Logs
 - Where does stdout and stderr go?
 - How long should a log exist?

`docker logs [OPTIONS] <Container>`

Docker Logs Options

Name, shorthand	Default	Description
<code>--details</code>		Show extra details provided to logs
<code>--follow , -f</code>		Follow log output
<code>--since</code>		Show logs since timestamp (e.g. 2013-01-02T13:23:37) or relative (e.g. 42m for 42 minutes)
<code>--tail</code>	all	Number of lines to show from the end of the logs
<code>--timestamps , -t</code>		Show timestamps
<code>--until</code>		API 1.35+ Show logs before a timestamp (e.g. 2013-01-02T13:23:37) or relative (e.g. 42m for 42 minutes)

Cleanup

- Docker Repository Cleanup
- Docker Registry Cleanup

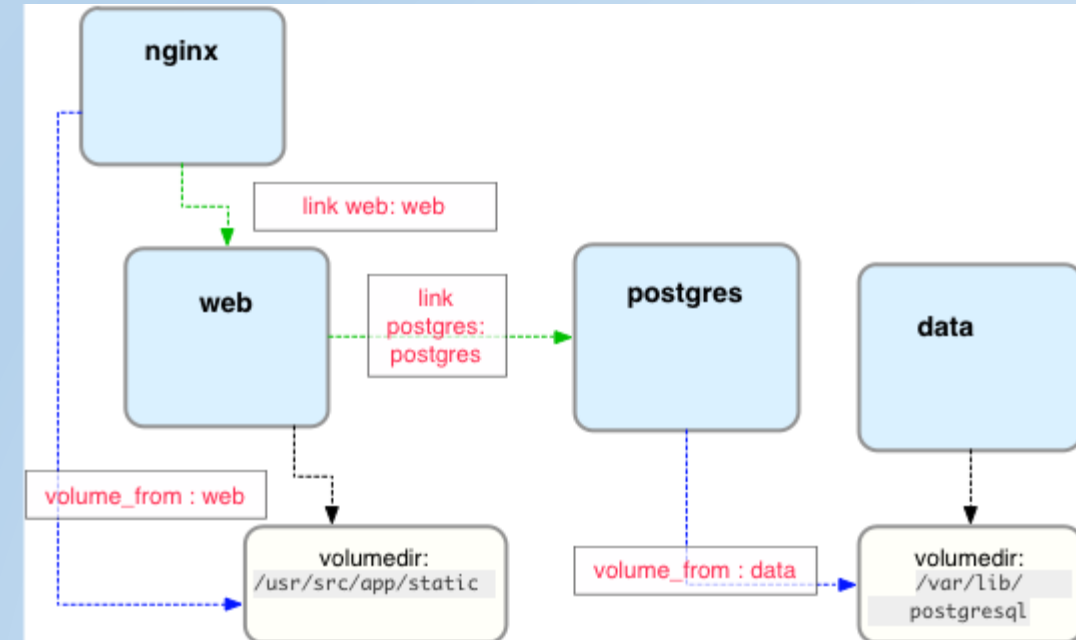
Some of Gerry's Scripts

- bcp: Build - Tag - Push
- dr: Docker Run
- rdi: Remove Docker Images
- drt: Tag
- sdi: Show Docker Images
- rdx: Remove Docker Images
- rdx: Remove Docker Images
- drs: Docker Registry Show
- drm: Docker Registry Manifest
- drx:
- drgcdr:

Docker Compose

- http://containertutorials.com/linked/docker_compose.html

- Remember Flask servers are simple
 - Examples
 - Docker Compose with 2 ports/base



- <http://containertutorials.com/docker-compose/spring-boot-app.html>

Examples

- From

1. Create a Dockerfile with the following content

```
FROM debian:wheezy

RUN apt-get update && apt-get install -y cowsay fortune
```

2. Go to the directory containing Dockerfile and execute the following command to build an image

```
$ docker build -t test/cowsay-dockerfile .
```

You will see output as shown below

```
Sending build context to Docker daemon 2.048 kB
Sending build context to Docker daemon
Step 0 : FROM debian:wheezy
wheezy: Pulling from debian
7a3e804ed6c0: Pull complete
b96d1548a24e: Already exists

Status: Downloaded newer image for debian:wheezy
---> b96d1548a24e
Step 1 : RUN apt-get update && apt-get install -y cowsay fortune
---> Running in 4404353a3643
Get:1 http://security.debian.org wheezy/updates Release.gpg [1554 B]
Get:2 http://security.debian.org wheezy/updates Release [102 kB]
Get:3 http://httpredir.debian.org wheezy Release.gpg [2390 B]
.....
Setting up perl (5.14.2-21+deb7u2) ...
update-alternatives: using /usr/bin/prename to provide /usr/bin/rename
---> ca3618d10f2a
Removing intermediate container 4404353a3643
Successfully built ca3618d10f2a
```

3. Check that image has been created

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL SIZE
test/cowsay-dockerfile	latest	ca3618d10f2a	3 minutes ago	126.9 MB
docker-dev	dry-run-test-2	db155754d7fc	6 days ago	1.571 GB
<none>	<none>	b01392d005bb	6 days ago	1.571 GB
debian	wheezy	b96d1548a24e	7 days ago	84.97 MB
debian	latest	df2a0347c9d0	7 days ago	125.2 MB
dockerswarm/dind-master	latest	bb4cd757411e	7 days ago	159 MB
<none>	<none>	f672d2db20f6	7 days ago	1.571 GB
<none>	<none>	1fe07c1fdf52	8 days ago	1.571 GB
dockerswarm/swarm-test-env	latest	01e6a0da0825	2 weeks ago	515.5 MB
ubuntu	14.04	07f8e8c5e660	3 weeks ago	188.3 MB
hello-world	latest	91c95931e552	5 weeks ago	910 B
busybox	latest	8c2e06607696	5 weeks ago	2.433 MB

4. Run the cowsay program using the built image

```
$ docker run test/cowsay-dockerfile /usr/games/cowsay "Hi!"
```

This will execute and show the output

```
< Hi! >
  -----
      \   ^__^
       (oo)\_____)
        (_____)  )\
           ||----w |
           ||     ||
```

5. Removing a Docker Image : Docker image can be removed using the following command

```
$ docker rmi test/cowsay-dockerfile
```

Overview of Kubernetes

- Declarative System (vs Imperative System)
 - What I want
 - Where I am
 - Important events
- Primary Services
 - Master Node
 - Worker Node
 - Kubelet
 - etcd
- Controllers & Operators

Understanding Kubernetes Objects

- There are Many Confusing Kubernetes Objects

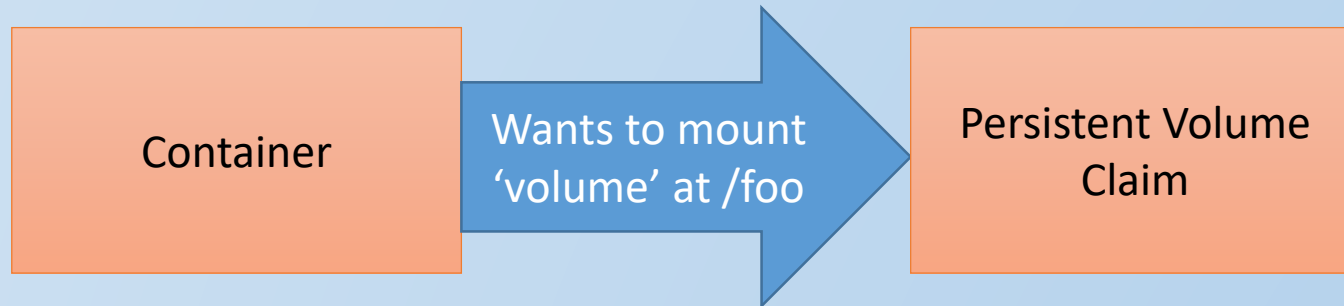
- Pod
- Deployment, StatefulSet, DaemonSet
- Volume
- PersistentVolumeClaim
- PersistentVolume
- PersistentVolumeAttachment
- StorageClass
- CSIDriver
- Oh my...

\$> kubectl api-resources

Where are the Containers?

- Pod
 - One or more Containers Running with a shared 'localhost'
- Deployment
 - A Template for deploying several Pods over multiple nodes (interchangeably)
- StatefulSet
 - A Template for deploying several Pods over multiple nodes (sticky id)
- DaemonSet
 - Template for running several Pods one per Node (or subset of Nodes)
- Others: ReplicaSet, CronJob, Job

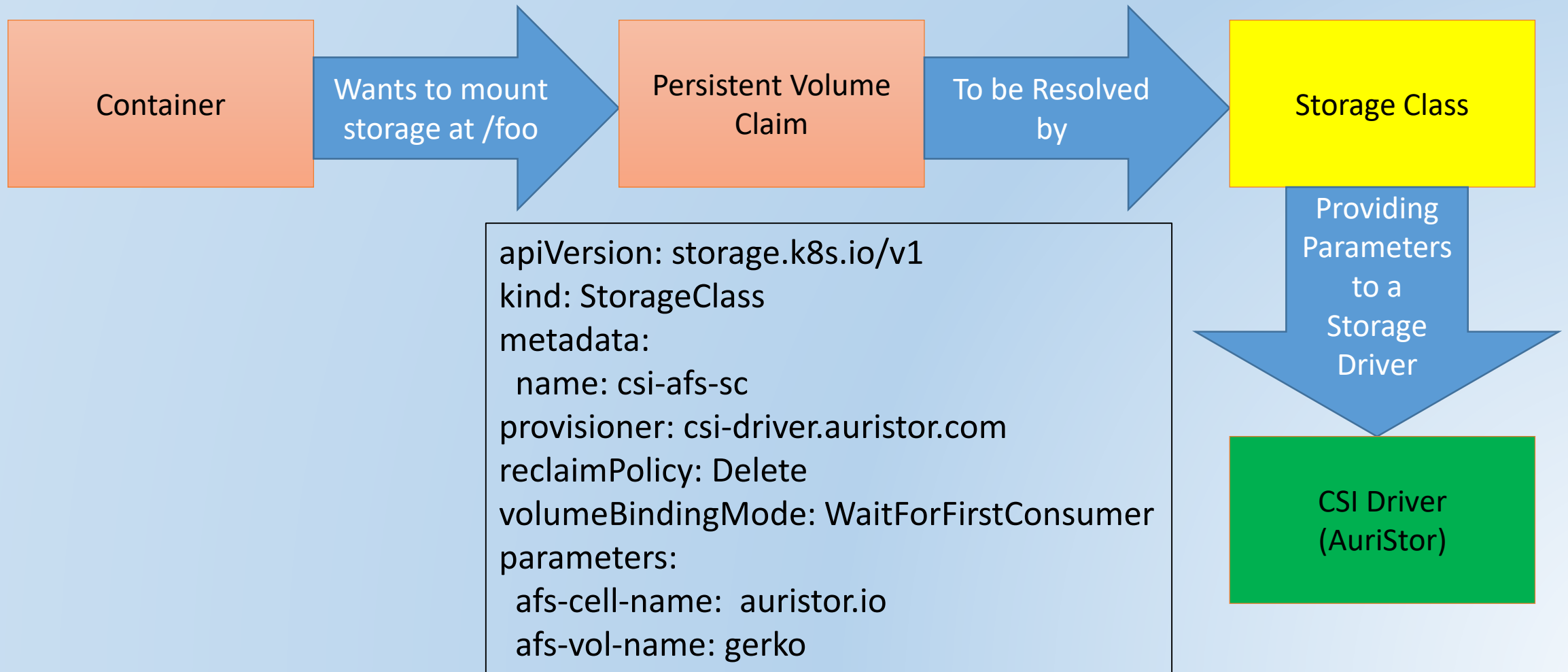
Not Quite Simplest Example



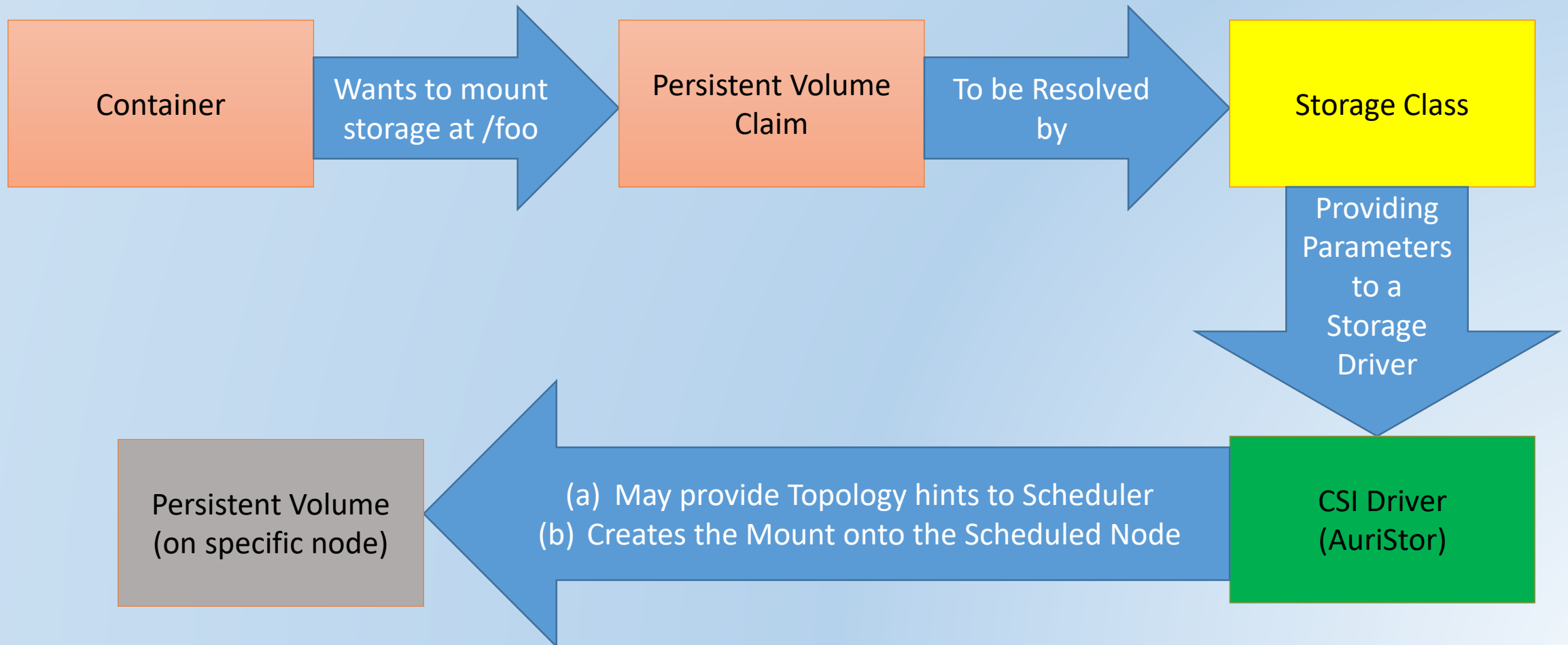
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-afs-sc
```

```
kind: Pod
apiVersion: v1
metadata:
  name: my-csi-app
spec:
  containers:
    - name: my-frontend
      image: busybox
      volumeMounts:
        - mountPath: "/data"
          name: my-csi-volume
      command: [ "sleep", "1000000" ]
  volumes:
    - name: my-csi-volume
      persistentVolumeClaim:
        claimName: csi-pvc
```

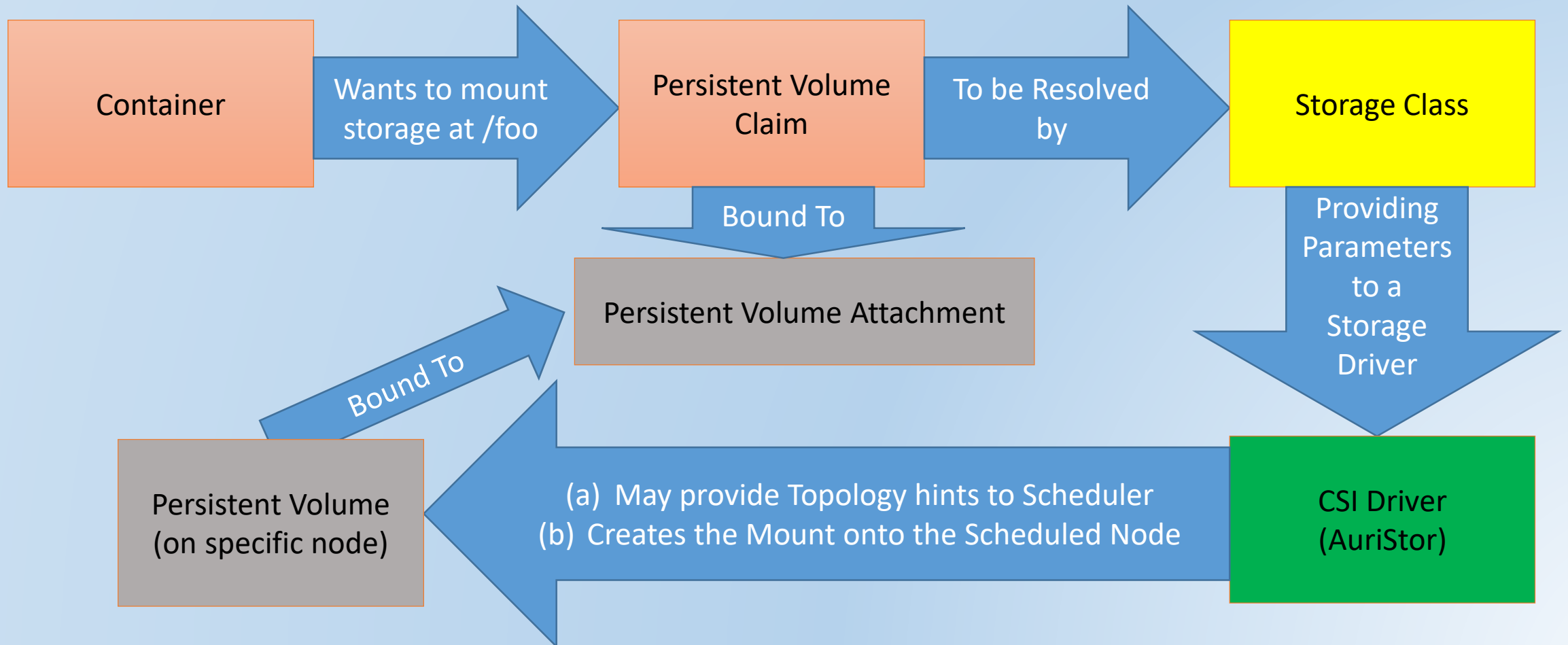
What the Cluster (Storage) admin Specifies



What the Kubernetes CSI Driver Does



What the external-attacher does



The Storage Class Parameters to CSI Driver

- AuriStor Driver Specific
 - Volume:
 - Existing Cell + Volume
 - Scratch + Quota
 - Topology
 - Don't care where the volume is
 - Schedule 'near' a File Server (ie Rack) with Volume (Replicate if necessary)
 - Do not Start until replication completes
 - Start immediately if none, but also start replication

Kubernetes Objects

- `o:\tech\kubernetes\Kubernetes-API-Objects.xlsx`

Network Objects

- Endpoints / EndpointSlice
- Service
 - An abstract way to expose an application running on a set of Pods as a network service.
- Ingress
 - Ingress can provide load balancing, SSL termination and name-based virtual hosting.

Security Objects

- ServiceAccount
- ClusterRole
- ClusterRoleBinding
- Secret
- Role
- RoleBinding

Persistent Storage Objects

- Volume
- PersistentVolumeClaim
- PersistentVolume
- PersistentVolumeAttachment
- StorageClass
- CSIDriver

Setting Up Kubernetes Cluster with kubeadm

- <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>

Kubernetes Guides

- [Kubernetes Concepts](https://kubernetes.io/docs/concepts/)
 - <https://kubernetes.io/docs/concepts/>
- [Kubernetes Tasks](https://kubernetes.io/docs/tasks/)
 - <https://kubernetes.io/docs/tasks/>
- [Kubernetes Tutorials](https://kubernetes.io/docs/tutorials/)
 - <https://kubernetes.io/docs/tutorials/>

Distributed Systems Study Group

First Meeting: Dec 3rd, 2019

Questions?

Gerry Seidman
gerry@iamx.com