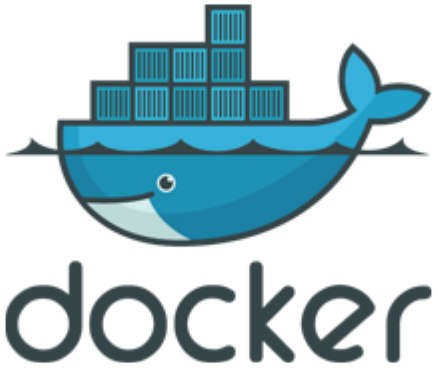
First Steps with:



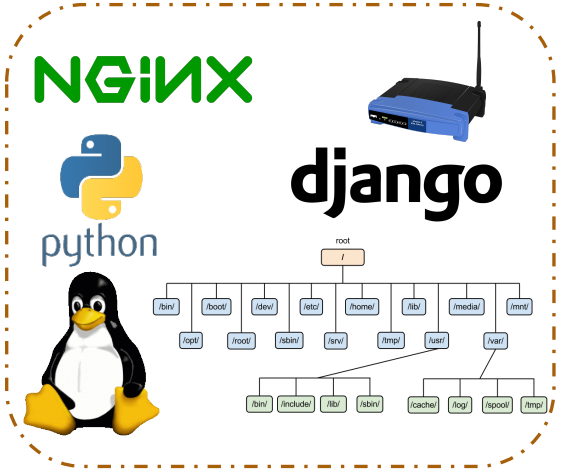
OCTOBER 11, 2017

"Old World" Problems

- **Rickety Software Delivery Pipelines**
  - *Different App Behavior in DEV, QA, PROD*
- **Poor Resource Utilization**
  - *Only a Few Applications per Machine*
- **Painful Software Administration**
  - *Discovery…. Installation…. Dependencies*

Solution – Containers!

CONTAINERS ARE:



- **<u>Isolated</u> Processing Environments**
  - *Can include their own network, storage, etc.*
  - *Host system houses containers*
- **Purpose: To Run an Application or Service**
  - *Includes EVERYTHING – (Small) OS…. Files…. Dependencies*
- **Lightweight**
  - *Minimum requirements to run*
  - *FAST!!!*

Containers – Why Use

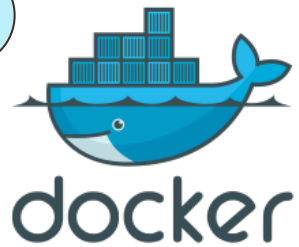
- Predictable **DEPLOYMENT** Environments
  - *DEV → QA → UAT → PROD*
  - *Deployment Types: (1) Blue/Green (2) Rolling (3) Phoenix*

- Efficient **PRODUCTION** Environments
  - Run More Applications on a Machine
  - Manage Highly-Elastic Applications

- Effective **DEVELOPMENT** Environments
  - *Easy Software Install and Admin / Configuration*
  - *Speedy Sandboxes*

Containers Landscape

1



- *Leading Container Platform*
- *Est. 2013*

2

Linux Containers

- *Formerly, LXC*

3

(Windows Containers)

- *Windows 10*
- *Windows Server 2016*

4

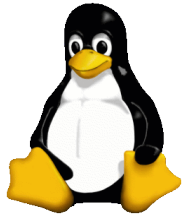
- *Container-Specific Operating System*
- *New Name: "Container Linux"*

5

- *Industry Standards*
- *Avoid Vendor Lock-In*

What Is Docker?

- **Creates, Manages and Orchestrates Containers**
  - *Separate Applications from Infrastructure*
  - *Better… Faster Software Deployment*



- **Packages an Application → Shippable (and Runnable) Image**
  - *"Build, ship, and run any app, anywhere"*
  - *Applications work the same on everyone's machines*

- **Docker:**
  - *"Dock Worker" moves goods into/out of ships*

What Is Docker?

3 Faces of Docker:

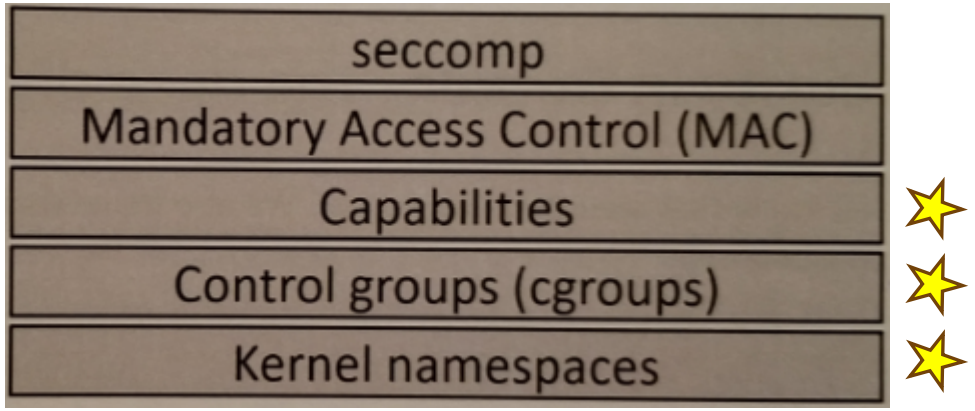
- **Docker (The Company – Docker, Inc.)**
  - *Formerly, a PaaS Provider called dotCloud*
  - *In 2013, Pivoted to Focus on its Container Software*

- **Open-Source Project (Renamed, "Moby")**
  - *Make Docker More Modular and Transparent*

- **Run-time and Orchestration Engine**
  - *3rd Party Products Plug into the Docker Engine*
  - *Community (Free) and Enterprise (Paid) Editions*

Enabling Technologies

- **Kernel Namespaces**
  - *Slice up an OS to look like multiple isolated environments (Process, Network, File System)*
- **Control Groups**
  - *Limit Resource Usage (CPU, RAM, Disk I/O)*
- **Capabilities**
  - *Choose which ROOT powers a container needs*



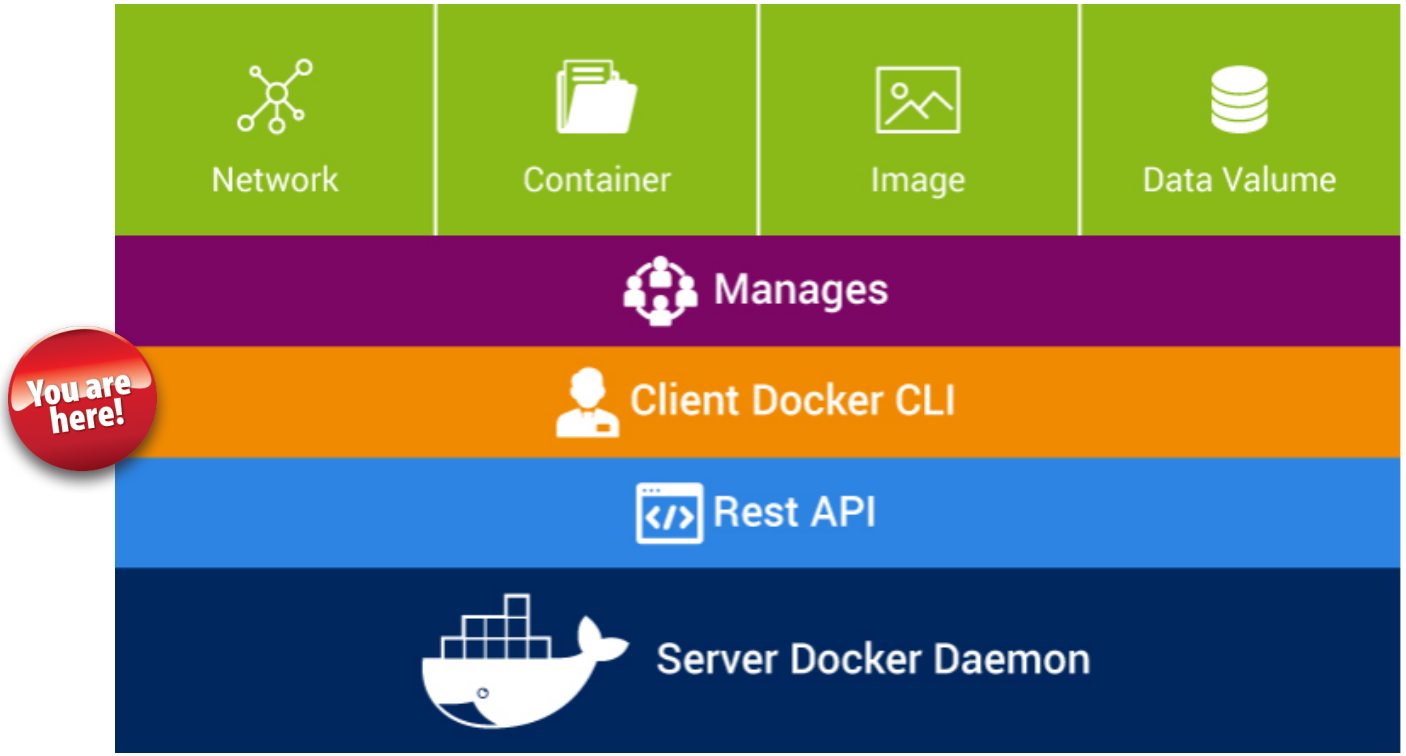
Why Did Docker "Win"?

- **Software Administration Lifecycle…. *EASY!***
- **Docker Hub and Official Registries**
  - *Establishes trust in images*
- **Modular and Platform-Agnostic Design**
  - *"Batteries Included… But Removable"*
- **Free and Open Source**

To The Terminal!

1. GENERAL

Docker – Architecture

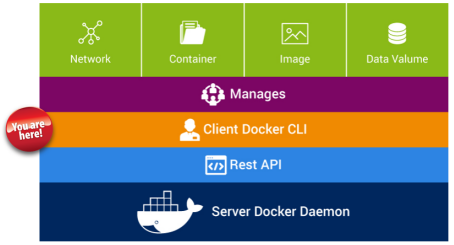


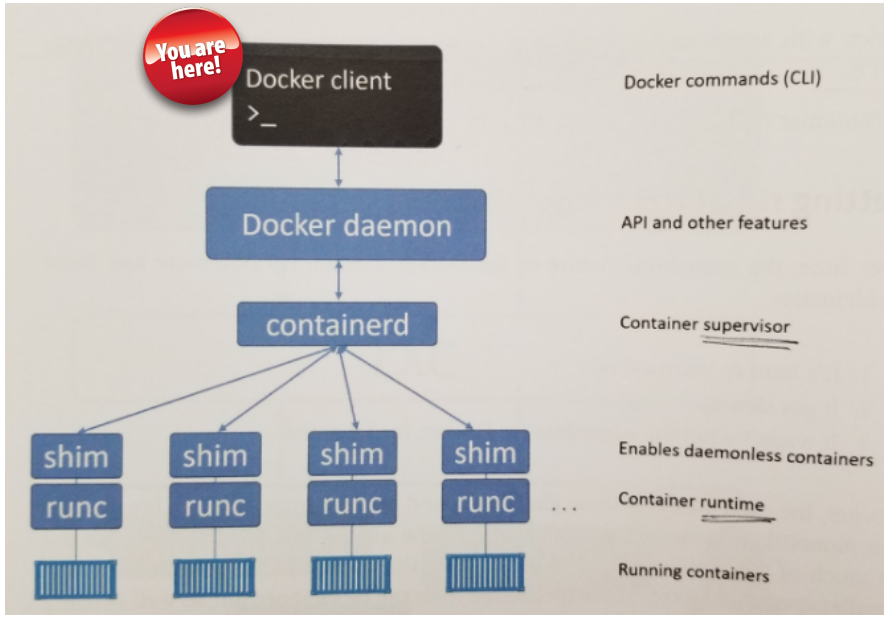
Docker Concepts

Engine	Coordinates all of the Docker components
Images	Core artifact of Docker…. Binaries (“Build”)
Containers	Launched from images …. Running applications (“Deploy”)
Dockerfiles	Readable code for image creation (“Source Code”)
Registries	Collection of repositories (which store images)

Engine

- Docker Client
  - *CLI tool that invokes Docker API calls*
- Docker daemon
  - *Formerly a Monolith… Now, just the Docker API*
- containerd
  - *Bridge b/w (1) Docker daemon and (2) runc*
  - *Container Lifecycle*
- runc
  - *Single Purpose → Create Containers*

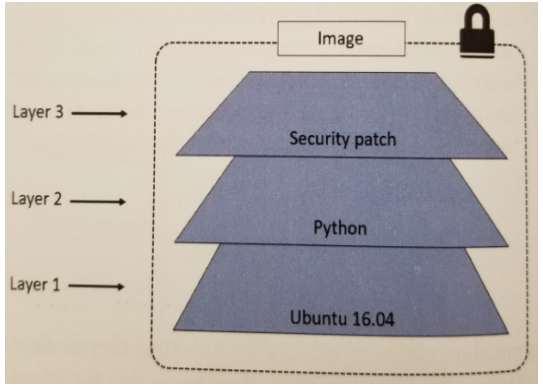




Images

- Application Packaged in a Standard, Portable File
  - *(1) Small OS …. (2) And ALL Files to Run App …. (3) Metadata*
- Stack of Read-Only Layers (Crypto Hashed)
- PULL Images onto Your Docker Host
  - *OR… Create Images from (a) Dockerfiles or (b) Containers*
- "Build-Time" Artifacts

- *docker image **pull***
- *docker image **ls***
- *docker image **rmi***
- *docker image **tag***



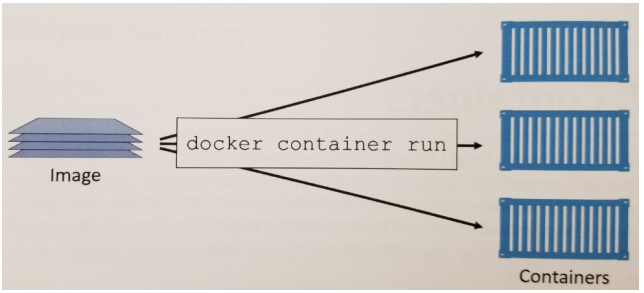
To The Terminal!

2. IMAGES

Containers

- Isolated Group of Processes
  - *Private Root File System…. Network Stack…. Process Tree*
  - *Unaware They Are Containers*
- Spawned from Images
- Run Until They Finish Executing
- “Run-Time” Artifacts

- *docker **create***
- *docker **start***
- *docker **stop***
- *docker **run***
- *docker **exec***
- *docker **rm***



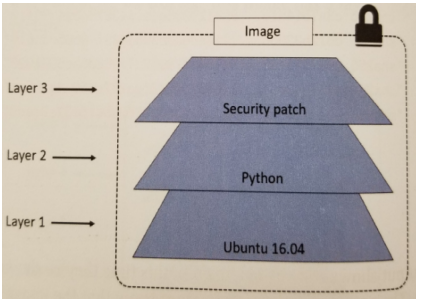
To The Terminal!

Containers – Easy Software Admin.

Traditional Software	Docker Equivalent	Docker Command
Find software	Docker Hub	
Download software, i.e. a zip file or MSI	Pull an image	docker pull
Install software	Create a container from an image	docker create
Start software	Run the container	docker start
Stop software	Stop the container	docker stop
Uninstall software	Remove the container	docker rm
Not Possible	Do all of this with one command!	docker run

Dockerfiles

- Recipes for **Building Images**
  - *Instructions and Shell Commands*
- First Instruction Specifies a Base Image
- Each Command Creates a New Layer
- Source Code Artifact
  - *Store in Version Control!!*



New Image Layers

0BD2 6C7D 2CDB FE3D 79A9
FD8E EE82 90B1 FB99 8B41

7D1B 91C0 F8BA CC25 8FCB
CBA3 9D42 4FB4 1B67 C696

F04D 825D FFE4 77C6 662B
82FA 8B8D 8A6C 72EA D091

22EC F97C DB2C 0488 91DC
A544 EC82 4F8C 21B4 1337

```
FROM debian:jessie
MAINTAINER NGINX Docker Maintainers "docker-maint@nginx.com"
ENV NGINX_VERSION 1.10.3-1~jessie
RUN apt-get update \
    && apt-get install -y ca-certificates nginx=${NGINX_VERSION} \
    && rm -rf /var/lib/apt/lists/*
# forward request and error logs to docker log collector
RUN ln -sf /dev/stdout /var/log/nginx/access.log \
    && ln -sf /dev/stderr /var/log/nginx/error.log
EXPOSE 80 443
CMD ["nginx", "-g", "daemon off;"]
```

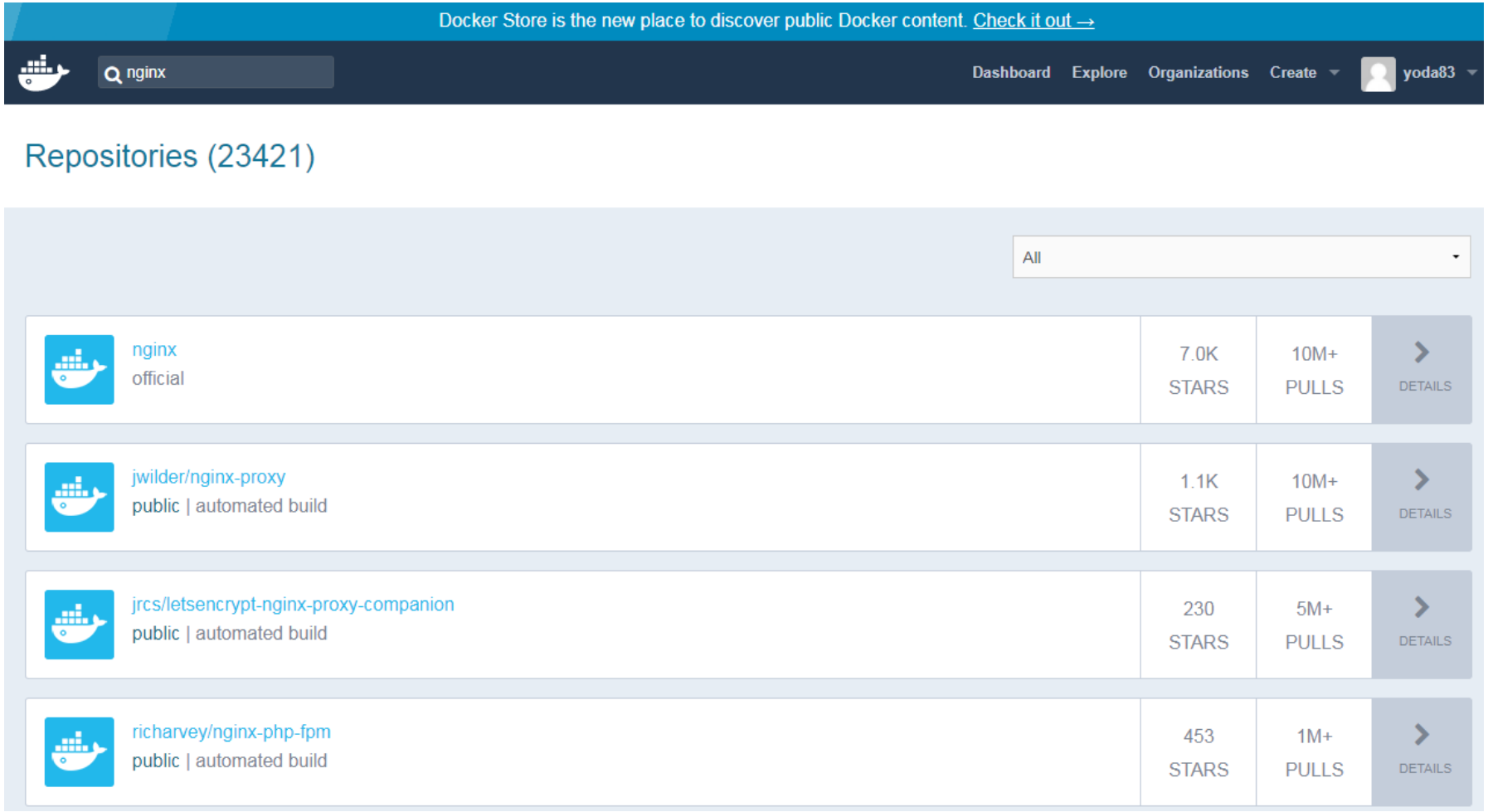
To The Terminal!

4. DOCKERFILE

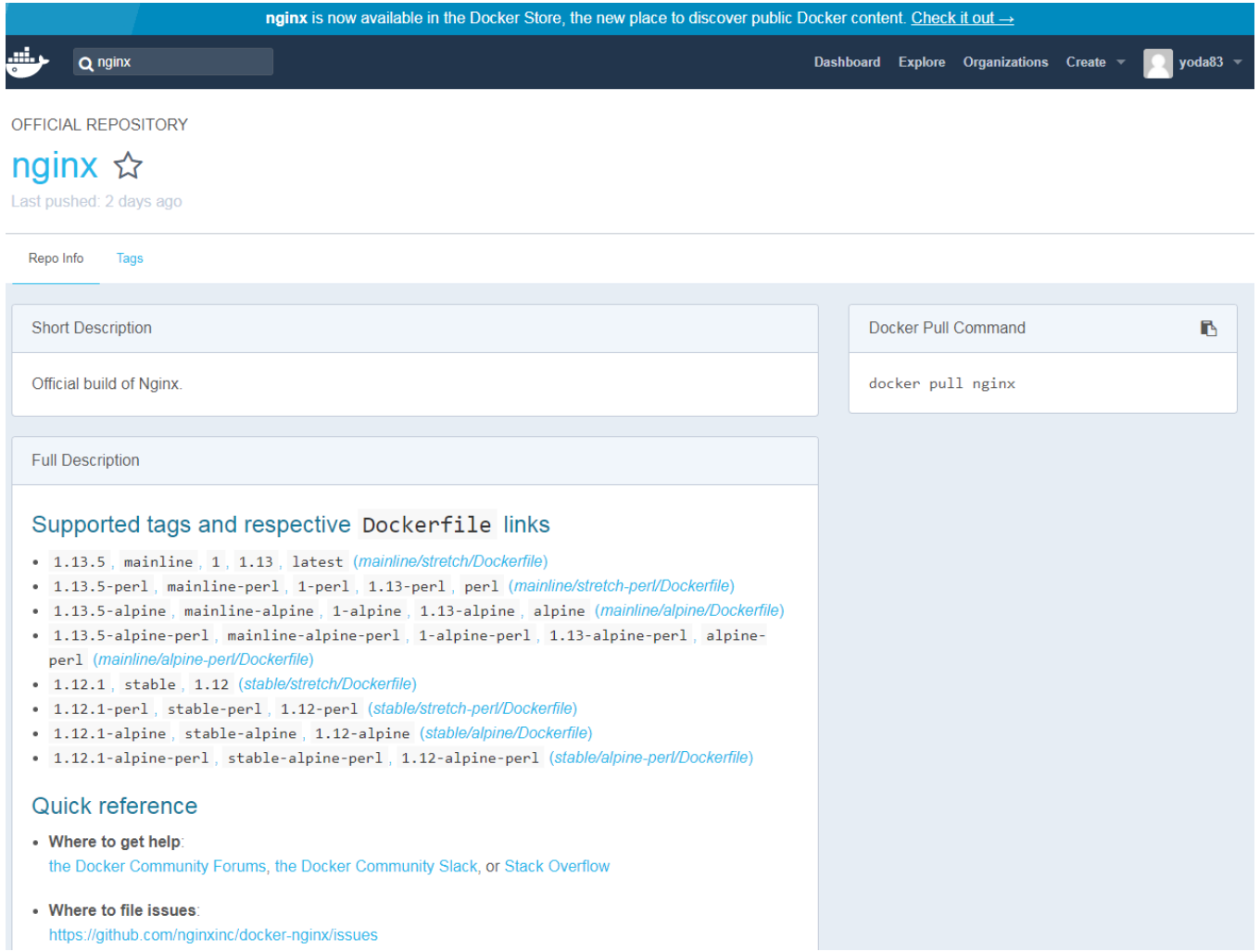
Registries

- Index of Docker Images
  - *Accessible by dockerd via HTTP*
- Docker Hub = Hosted Registry by Docker, Inc.
  - *Public and Private Repositories*
  - *Official vs. No-name*
- Other Public Registries Include:
  - *Amazon ECR*
  - *Google Container Registry*
  - *quay.io*
- Private Registries for Sensitive Images

Registries


Docker Store is the new place to discover public Docker content. Check it out →
nginx
Dashboard
Explore
Organizations
Create
yoda83
Repositories (23421)
All
nginx
official
7.0K STARS
10M+ PULLS
DETAILS
jwilder/nginx-proxy
public | automated build
1.1K STARS
10M+ PULLS
DETAILS
jrcs/letsencrypt-nginx-proxy-companion
public | automated build
230 STARS
5M+ PULLS
DETAILS
richarvey/nginx-php-fpm
public | automated build
453 STARS
1M+ PULLS
DETAILS

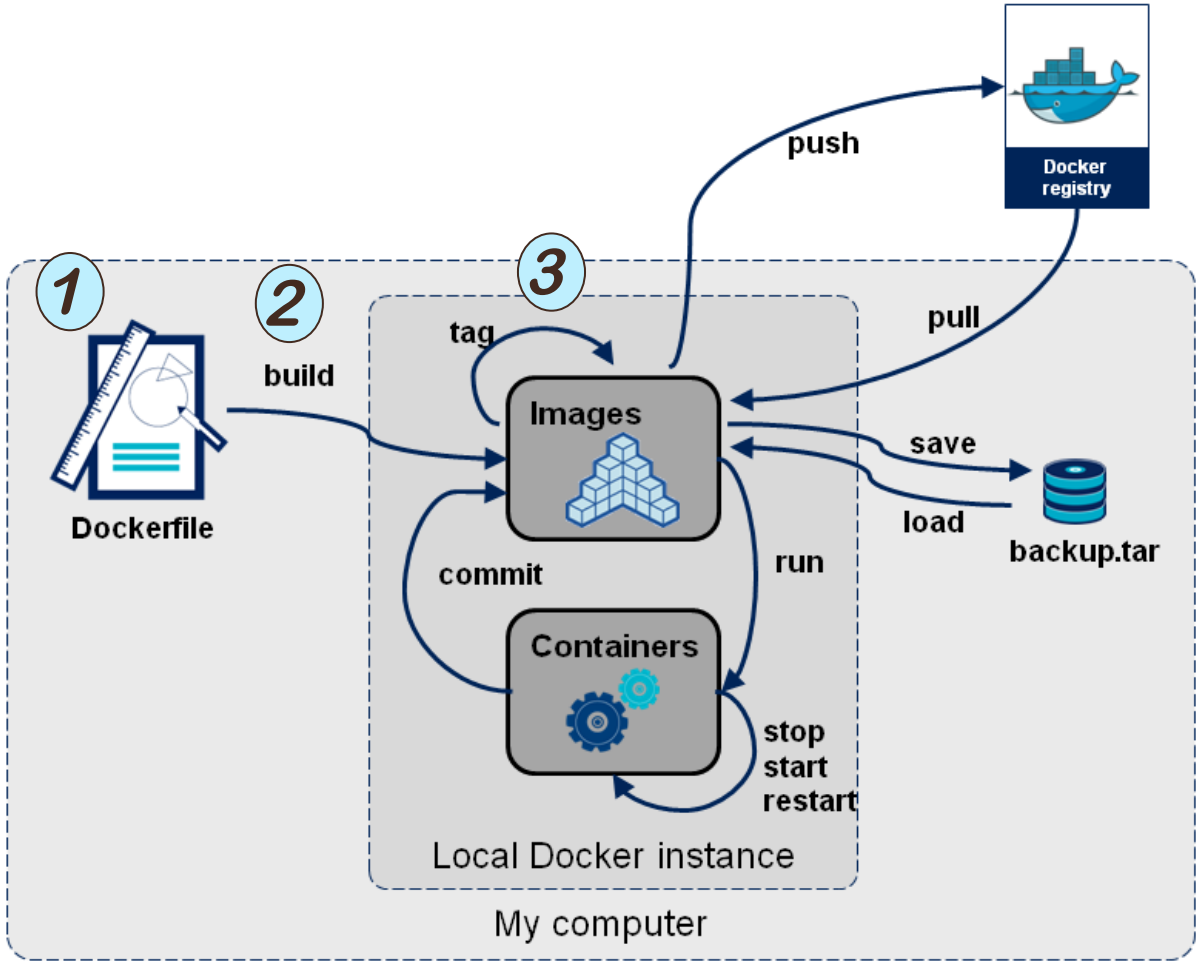
Registries


nginx is now available in the Docker Store, the new place to discover public Docker content. Check it out →
nginx
Dashboard
Explore
Organizations
Create
yoda83
OFFICIAL REPOSITORY
nginx
Last pushed: 2 days ago
Repo Info
Tags
Short Description
Official build of Nginx.
Docker Pull Command
docker pull nginx
Full Description
Supported tags and respective Dockerfile links
1.13.5, mainline, 1, 1.13, latest (mainline/stretch/Dockerfile)
1.13.5-perl, mainline-perl, 1-perl, 1.13-perl, perl (mainline/stretch-perl/Dockerfile)
1.13.5-alpine, mainline-alpine, 1-alpine, 1.13-alpine, alpine (mainline/alpine/Dockerfile)
1.13.5-alpine-perl, mainline-alpine-perl, 1-alpine-perl, 1.13-alpine-perl, alpine-perl (mainline/alpine-perl/Dockerfile)
1.12.1, stable, 1.12 (stable/stretch/Dockerfile)
1.12.1-perl, stable-perl, 1.12-perl (stable/stretch-perl/Dockerfile)
1.12.1-alpine, stable-alpine, 1.12-alpine (stable/alpine/Dockerfile)
1.12.1-alpine-perl, stable-alpine-perl, 1.12-alpine-perl (stable/alpine-perl/Dockerfile)
Quick reference
Where to get help:
the Docker Community Forums, the Docker Community Slack, or Stack Overflow
Where to file issues:
https://github.com/nginxinc/docker-nginx/issues

To The Terminal!

search

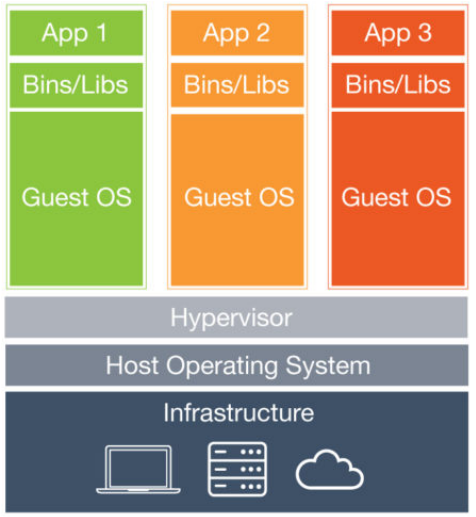
Docker Flow


Docker registry
push
pull
1
2
3
build
tag
Images
Dockerfile
save
load
backup.tar
commit
run
Containers
stop
start
restart
Local Docker instance
My computer

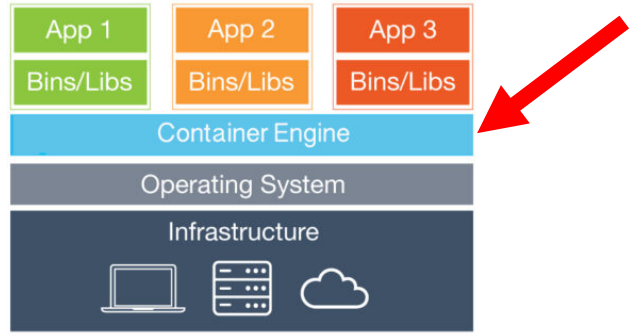
Containers vs. VM's

Virtual Machines

- *Rely on Hypervisor*
- *Carve up HARDWARE resources*
- *Bring their own Operating System*
- *Slower and Heavier than Containers*



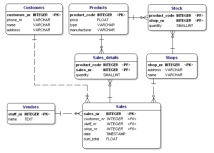
Hypervisor-based Virtualization



Container virtualization

Docker in the Software Stack

5. Custom Application ~ *Your Logic; Algorithms*



4. Base Configuration ~ *Software Tools Wired Up*

3. Base Application ~ *Software Tools (Generic)*

2. Custom Image ~ *"Hardened" AMI's*

1. Base Image ~ *Virtual Machines… AMI's*

To The Terminal!

6. PROCESS ISOLATION

Container Mgmt

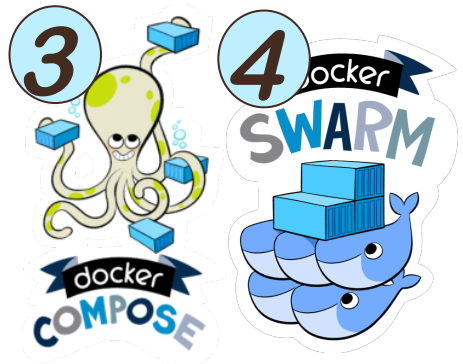
WHO MANAGES:

- Container Sprawl?
- Container Orchestration?
- Container Failure and Recovery?

Container Mgmt

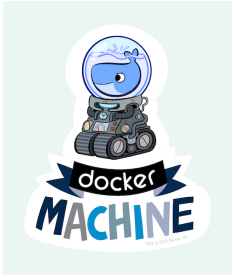
SERVICES INCLUDE:

- Initial Launch
- Service Discovery
- Fault-Tolerance
- Horizontal Scalability



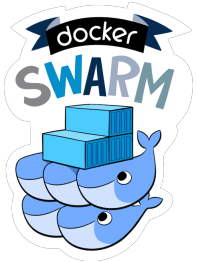
Docker Ecosystem

- Container Engine

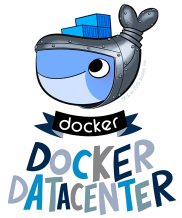


- Manage Virtual Docker Hosts Remotely

- Cloud-based Registry for Docker Images



- Cluster and Scheduling Tool for Docker Containers



- Helps Enterprises Deploy Their Own Docker Platforms

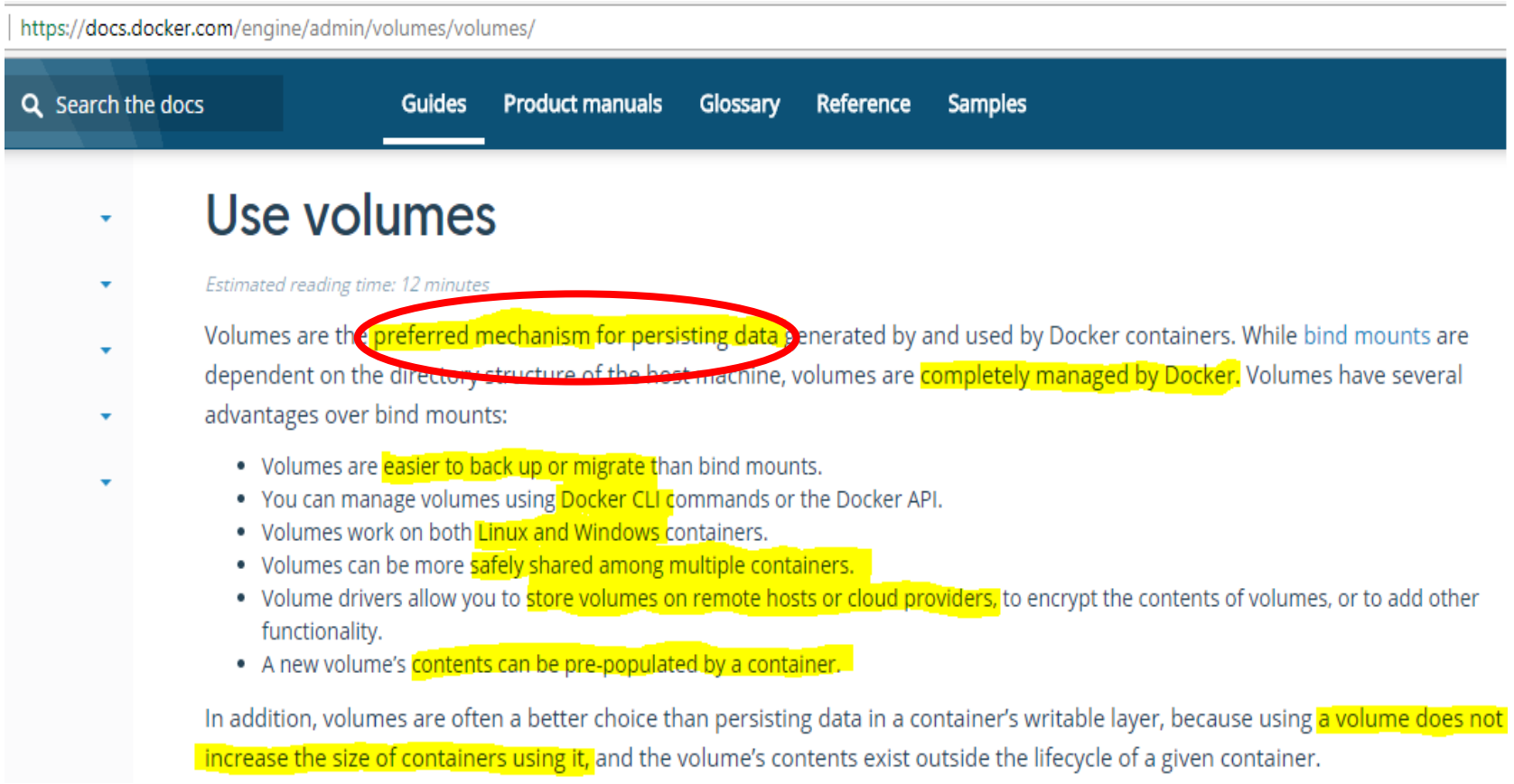
- For Trusted Commercial (and Free) Software

- Define and Run Multi-Container Docker Applications

- Cloud Platform with Docker as a First Class Citizen (Like AWS…)

- Storage and Content Delivery System (CDN) for Docker Images

Docker Volumes



Docker…. Reasons to Avoid

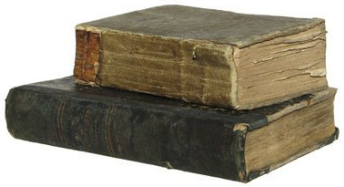
- No Support (Windows 2012 or earlier)
- Security and SLA's
- Lower-Level Infrastructure (e.g. AMI's)
- Young Platform (esp. on Windows)

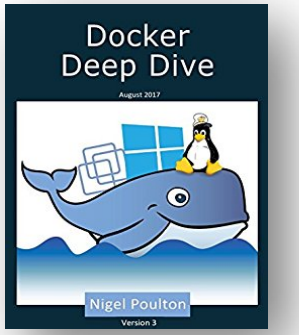
Next Steps

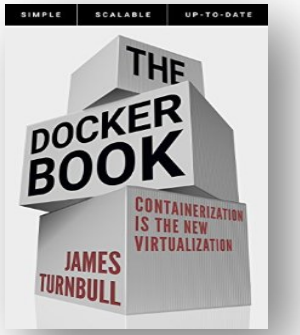
- Docker Volumes
- Dockerfile
- Docker Compose and Swarm
- Docker in a CI/CD Pipeline

Docker Resources

- https://app.pluralsight.com/library/courses/containers-images-big-picture/table-of-contents
- https://app.pluralsight.com/library/courses/docker-deep-dive/table-of-contents
- https://www.xenonstack.com/blog/docker-overview-a-complete-guide
- https://docs.docker.com/get-started







Thanks!