Building multi-SDK containers for all your API needs

Mike Nelson
Technical Evangelist @ Pure Storage
@mikenelsonio

PURESTORAGE®

Mike

mikenelsonio

nelmedia

mikenelson-io

- Almost 40 years in tech
- Principal Technical Evangelist @ Pure Storage
- Experience from Helpdesk to Architect
- Scripter, not a coder
- Passion for community, teaching, learning
- Beer, BBQ, & Gadgets

MVP Microsoft® Most Valuable Professional

Citrix Technology Advocate CTA

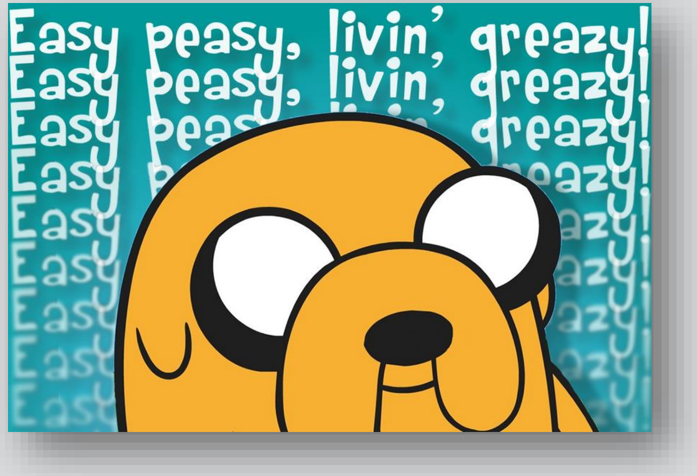
vmware® vEXPERT ★★★★★★★★

PURESTORAGE®

Why would we do this?

Is there a want or a need?

Do we want to make non-developer lives easier?



What SDKs do we use?

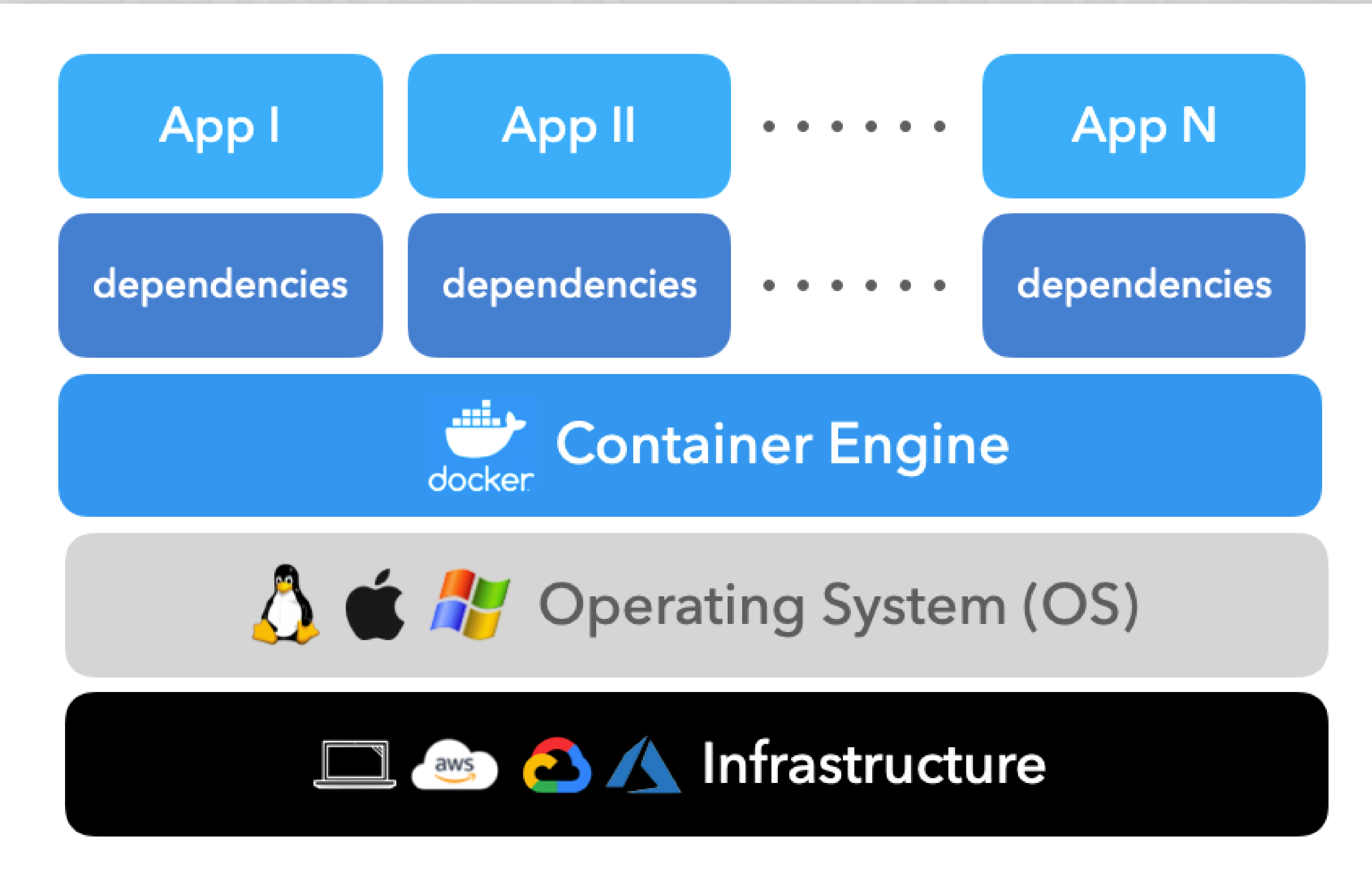
Python

Ansible

Terraform

pfctl <custom CLI>

Add swagger because, why not?



App I

App II

App N

dependencies

dependencies

dependencies

Container Engine

Operating System (OS)

Infrastructure

Docker

The basics to know

- Multi-platform & Docker Desktop (meh)
- Images - A blueprint from which an arbitrary number of new containers can be started.
- Port-forwarding – expose container ports
- Volumes – persistent data for containers
- docker-compose – A tool to define & run multi-container Docker applications
- docker build(x) - builds Docker images from a Dockerfile and a “context” (path, URI)
- Dockerfile - A set of precise instructions, stating how to create a new Docker image, setting defaults for containers being run based on it, & more. Best case, it’s going to create the exact same image for running it at any point in time.
- Repository

Dockerfile

- **ADD** copies the files from a source on the host into the container's own filesystem at the set destination.
- **CMD** can be used for executing a specific command within the container.
- **ENTRYPOINT** sets a default application to be used every time a container is created with the image.
- **ENV** sets environment variables.
- **EXPOSE** associates a specific port to enable networking between the container and the outside world.
- **FROM** defines the base image used to start the build process.
- **MAINTAINER** (deprecated – use LABEL) defines a full name and email address of the image creator.
- **RUN** is the central executing directive for Dockerfiles.
- **USER** sets the UID (or username) which is to run the container.
- **VOLUME** is used to enable access from the container to a directory on the host machine.
- **WORKDIR** sets the path where the command, defined with CMD, is to be executed.
- **LABEL** allows you to add a label to your docker image.

Building It

- Keep it simple.
- Use a small footprint to start - Alpine Linux (~8MB), Ubuntu (~700MB), Nano(~295MB)/Server Core (~2.76GB)
- Use a dockerfile and docker build(x)
- Mount volumes for persistent dynamic scripts & data
- Copy in necessary files on build
- Automate the build with a GitHub Action workflow – CI/CD pipeline
- Use quay.io for your containers, or host your own

demos & thank you!

@mikenelsonio
github: mikenelson-io
email: mike@mikenelson.io