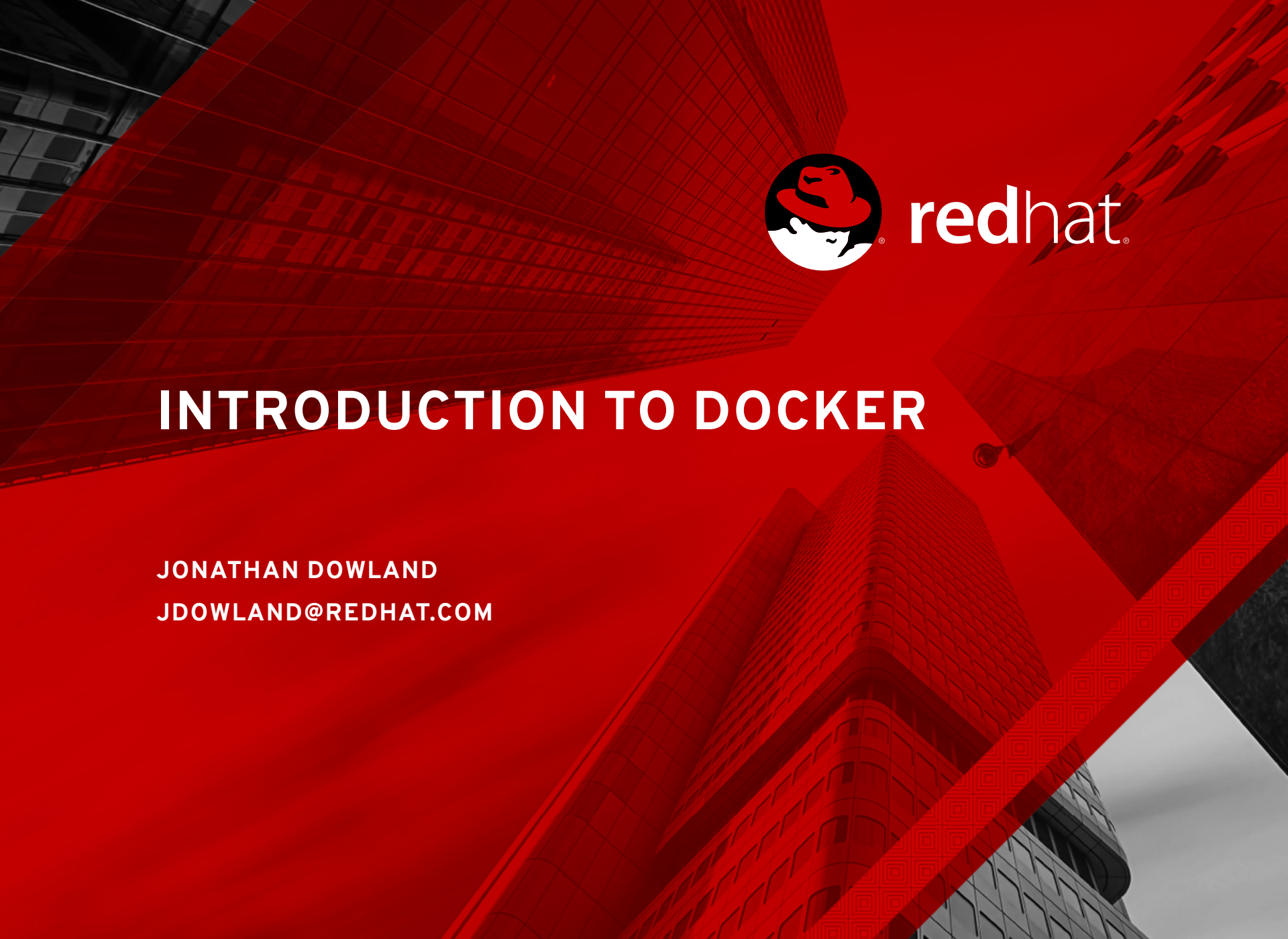


INTRODUCTION TO DOCKER

JONATHAN DOWLAND

JDOWLAND@REDHAT.COM

ABOUT ME

- Senior Software Engineer, Cloud Enablement, Red Hat
- Formerly CS Support Team Leader (2010-2015)
- Open source stuff (Debian etc.)

also a guest member of staff with the School (working on CS history/software & hardware preservation)

It's been a while since my last lecture so I might be a bit rusty, be gentle!

DOCKER IS...

"A SOFTWARE CONTAINERIZATION PLATFORM"

Note the verb here: containerization. Not just using containers, but creating them too.

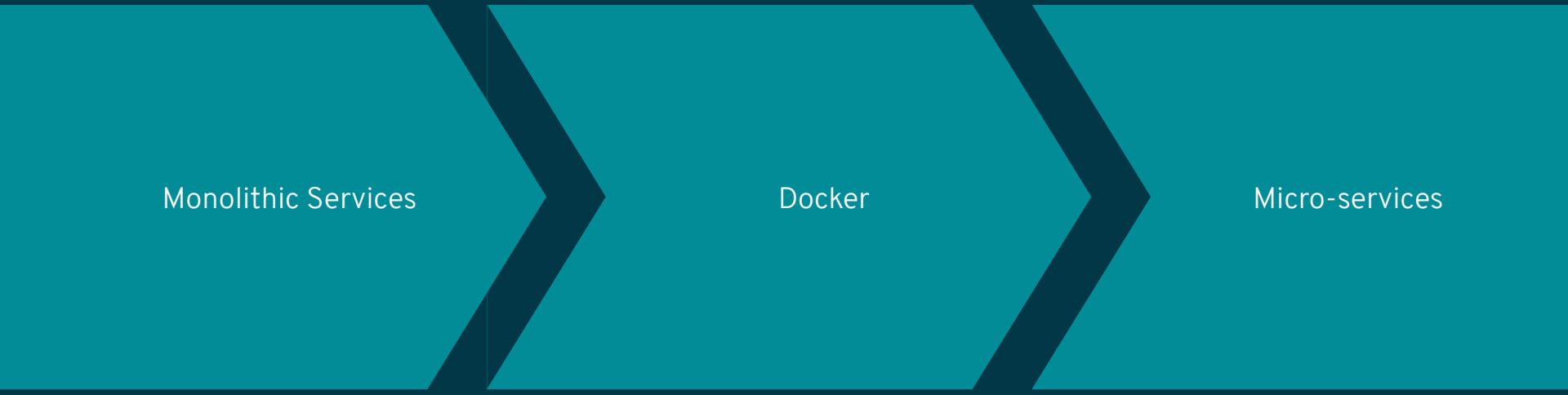
TERMINOLOGY

- Image
- Container

Image: the static things you create, share, upload/download from registries

container: a running instance of a container

MICROSERVICES



moving towards a microservices architecture

microservices loosely defined, but philosophically: small services, doing one thing and doing it well, communicating amongst themselves using open protocols (no privileged channels), such as APIs and/or messaging systems

docker is pragmatic in that it lets you build using Tomorrow's architecture (microservices) using today's components (existing services, servers, operating systems)

DOCKER IS…
A PACKAGING FORMAT

Packaging systems have been around a LONG time

RPM, DEB etc., decades old

but packaging is not a solved problem (or we'd stop inventing new ones, there must be a new packaging scheme invented every week)

A docker image is essentially a package of software, some dependencies, and some metadata.

DOCKER(HUB) IS...
A REPOSITORY OF IMAGES

docker-dot-com run the docker hub, a public repository of lots of software. you can push your own

the docker registry software is itself distributed as a docker image

you can run your own (basic) one trivially

Red Hat use a registry to distribute images of its enterprise products to customers

DOCKER IS…

PROCESS ISOLATION

you can isolate processes from each other inside containers, so they cannot interfere with each other; you can isolate files on your host machine or in other containers from each other; you can isolate containers from each other and the world at large from a networking perspective; you can apply resource limits to prevent a container consuming all CPU/mem/etc. on a host

WHY DOCKER?
PACKAGING

from the What to the Why

a simple one - to distribute software, in an OS agnostic fashion

run SW in a docker container on multiple OSes, the same image in each case

package *your* software for consumption by anyone who can run docker, no need to learn a dozen or more packaging schemes

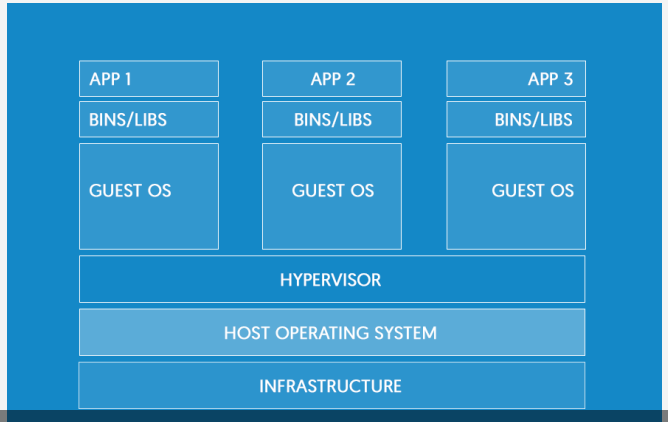
WHY DOCKER?

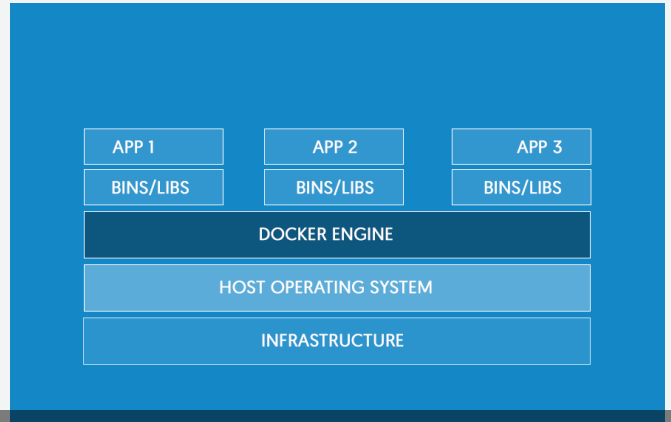
REPOSITOR(IES) OF IMAGES

follows from packaging; a way of distributing packages

WHY DOCKER?
UTILISATION

- lower overhead than virtual machines





slides taken from docker-dot-com's own "what is docker" material

VM model, copies of OS kernel and system processes running in each VM instance

docker model, those processes not running - hypervisor equivalent is the docker engine

some other container schemes don't need a docker engine equivalent at all either

WHY DOCKER?
ISOLATION

evaluated untrusted code in a (more) secure environment than your personal OS

mitigate (not eliminate) risk of malicious code, or bugs, or external hacks

defense in depth

ISOLATION

CASE STUDY: STEAM

real live bug, deleted all of a user's files if triggered

CAVEAT / DISCLAIMER / ETC

The following bug never reached the public, and is only public itself due to Red Hat's open development practices. It was found and fixed by our QA processes and is here to illustrate the general class of bug only!

ISOLATION

CASE STUDY: SQUID (IN RED HAT)

hypothetical bug (never released), deleted all files on system

(demo)

demo of starting a (any) container as root, running "rm -rf /*", demonstrating the lack of /usr /bin etc., inability to "ls" anymore, etc. -- host machine is fine

WHY DOCKER?
REPRODUCIBILITY

“A Framework for Scientific Workflow Reproducibility in the Cloud”

Rawaa Qasha , Jacek Cała, Paul Watson

of relevance to scientists

https://www.researchgate.net/profile/Rawaa_Qasha/publication/307905445_A_Framework_for_Scientific_Workflow_Reproducibility_in_the_Cloud/links/57ecf52c08ae92eb4d2689d0.pdf

this paper won best paper award at an eScience conference

published by your colleagues/supervisors!

earlier studies demonstrated that many publically available workflows, published as part of research, are not reproducible

study goes into much more depth than I can about why, but assumptions about environment is part of the problem

this paper outlines a system which uses docker as a component of a workflow system designed to be reproducible in the future

docker makes the environment (more) explicit, eliminating (some) of these problems

DEMO

- pre-requisites
- using a docker image
- building a docker image

demo notes/transcript available separately

CASE STUDY
IKIWIKI

- http://support.cs.ncl.ac.uk/

ikiwiki used to power support.cs.ncl.ac.uk

very flexible and powerful

a difficult piece of software to install and configure, lots of dependencies, some optional depending on features/plugins you wish to use

bad defaults (ugly/no theme, etc.)

(brief demo of a work-in-progress "ikiwiki-in-a-box" docker container that makes some decisions about configuration for you)

DOCKER FOR YOU

- evaluating software
- sharing work with colleagues/supervisors/collaborators

GOING FORWARD

- questions?
- jon@dow.land
- slides going up at https://jmtd.net/
- ☕

I'm in the building on the 7th floor

happy to take any questions via email (personal email best)

these slides will be up on my website

always happy to share a coffee and discuss things

redhat.

THANK YOU!

linkedin.com/company/red-hat

twitter.com/RedHatNews

facebook.com/redhatinc

plus.google.com/+RedHat

youtube.com/redhat