nic

nordic infrastructure conference

2017

The premium event for IT-professionals

Feb. 1-3rd in Oslo Spektrum

SQL Server in DevOps

Travis Wright
Principal Program Manager
SQL Server Engineering Team
Microsoft
@radtravis

nic 2017
nordic infrastructure conference

Agenda

- Background on SQL Server, DevOps, Containers, and Container Management Platforms
- Demos
- Lessons Learned
- What’s Next
- Get Involved

DevOps Principles in Focus

- Embrace emerging technology and patterns
- Challenge the status quo
- Agility, efficiency, and speed
- DevOps is a partnership between Dev and Ops
  - Ops enables and controls
  - Devs build and maintain apps and services
  - Both provide support

Containers

- Fad or fundamental transformation?

Container Benefits

All the benefits of VMs plus…

- **Reduced size on disk** – better hardware utilization
- **Reduced CPU/memory consumption** – better hardware utilization
- **Reduced deployment size** – faster deployments and scale up/down
- **Reduced patching** – less effort, less vulnerability, less down time

This is not a fad!

…but wait…

- “Containers are emphemeral”
- “Containers aren’t for databases”

Docker 🤍 Databases

- Many of the most popular images are databases
  - Postgres: 10M+ pulls
  - Mysql: 10M+ pulls
  - Redis: 10M+ pulls
  - Mongo: 10M+ pulls
- SQL Server on Linux has had 250K+ pulls in the first two months

Persisting Storage

- Mount a volume to the host
  - Local storage
  - Remote storage

```
docker run … -v /my/host/dir:/my/container/dir …
```

- Mount a container volume

```
docker create -v /mydata --name mydatacontainer …
docker run --volumes-from mydatacontainer …
```

Read this!
https://docs.docker.com/engine/tutorials/dockervolumes/

Build & Test Locally in Dev Environment

- Build locally on Windows, Linux, or macOS
  - Windows
    - Linux Docker containers using Docker for Windows
    - Windows containers on Windows 10 Anniversary Edition+
  - macOS
    - Linux Docker containers using Docker for Mac ← Use for demo today
  - Linux
    - Use Docker Engine natively
    - There are other container engines like LXC

Testing & Release

- Use Docker Engine natively ← Use for demo today
- Use a container management platform (aka “orchestrator”)
  - Kubernetes
  - Red Hat OpenShift (based on Kubernetes) ← Use for demo today
  - Docker Swarm
  - Mesosphere DC/OS
- Use a cloud service
  - Azure Container Service (ACS)
  - AWS EC2 Container Service (ECS)

Real World Example

SQL Server Engineering Team uses Kubernetes in Azure VMs for automated testing of SQL Server on Linux

- Automated build process creates the container image
- Extended existing test system to handle provisioning and test execution/targeting
- ~700 containers per test run, usually once per day
- 150 VM hosts in Azure; 128 GB/8 cores
- 20+ containers/VM in some cases
- High density, each SQL Server container listens on a different port

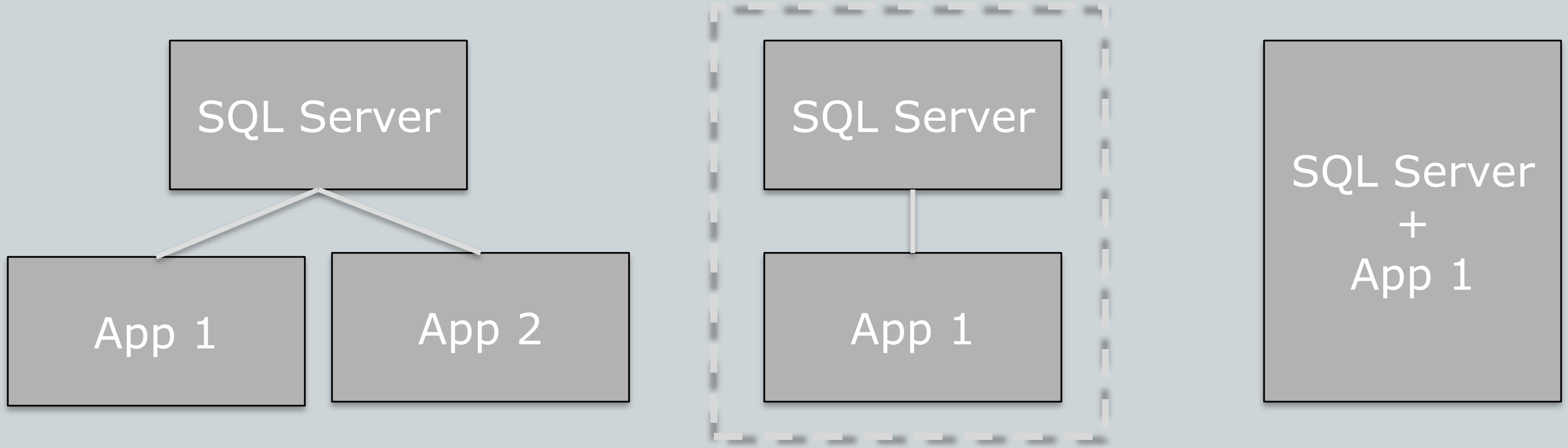
Key Docker Terminology and Commands

- **Image** – A definition. Defines what software is included and how it runs.
- **Container** – A running instance based on the image.

- `docker pull` – download an image from a Docker respository
- `docker run` – create a container from an image
- `docker ps` – list all locally running containers
- `docker images` – list all locally cached images

You do not “install” a Docker container! ☺

Application Deployment Patterns Using Containers



SQL Server

App 1

App 2

Centralized SQL Server

SQL Server

App 1

Docker Compose

SQL Server
+
App 1

Monolithic App

Methods for Deployment

- Deploy standard SQL Server container image.
- App deploys the DB at start up.

← Demos #1,2 today

- Use standard SQL Server image as a base layer. Include DB inside of image.
- At run time the CMD in the Dockerfile attaches/restores the DB.

- Use standard SQL Server image as base layer.
- At run time the DB is created by the CMD in the Dockerfile.

← Demo #3 today

Demo #1
Simple Example + Voting App

Demo #2
ASP.Net Docker-Compose App

Demo #3
Node Monolithic App

Lessons Learned

- Tools aren't in image. See examples for how to install them.
- Can't connect directly to the SQL Server from outside OpenShift network - no access to port 1433
- No way to create a DB simply at `docker run` time
- Need to code around delays to start, SQL Server takes too long to start
- SQL Server currently requires 3.25 GB of RAM – more than *should* be required
- Make sure you are running on latest Docker-Engine version (demo built using Docker 1.12.5) - ASP.Net container will have issues if not. Current OpenShift-All-In-One Vagrant box has older version of Docker Engine.
- Issue with the fact that the container runs as root. Open Shift doesn't allow that by default. You can change the perms, but it requires cluster administrator privs.
- Docker, OpenShift/Kubernetes, and SQL Server on Linux is still relatively new, especially to ops teams.
  - No established best practices
  - Minimal documentation
  - Few experts

What’s Next

- High availability in container management platforms
- Performance testing and tuning
- Reduce minimum RAM requirement and start up time
- Docker image improvements
  - https://github.com/Microsoft/mssql-docker/issues
- More testing of SQL Server on Windows containers → officially support
- General Availability later this year of SQL Server v.Next including support for SQL Server on Linux containers
- Make SQL Server container images available everywhere

Get Involved

- Get the SQL Server container images
  - hub.docker.com/r/microsoft/mssql-server-linux (v.Next)
  - hub.docker.com/r/microsoft/mssql-server-windows (v.Next)
  - https://hub.docker.com/r/microsoft/mssql-server-windows-express/ (2016 SP1)
  - Mesosphere DC/OS Universe image
- Provide feedback
  - https://github.com/Microsoft/mssql-docker/issues
- Get samples from today
  - https://github.com/twright-msft/mssql-node-docker-demo-app
  - https://github.com/twright-msft/mssql-aspnet-docker-demo-app
- Contact me if you want to work on cool stuff together!

Thank you!!

twright@microsoft.com
twitter.com/radtravis
linkedin.com/in/radtravis

Microsoft