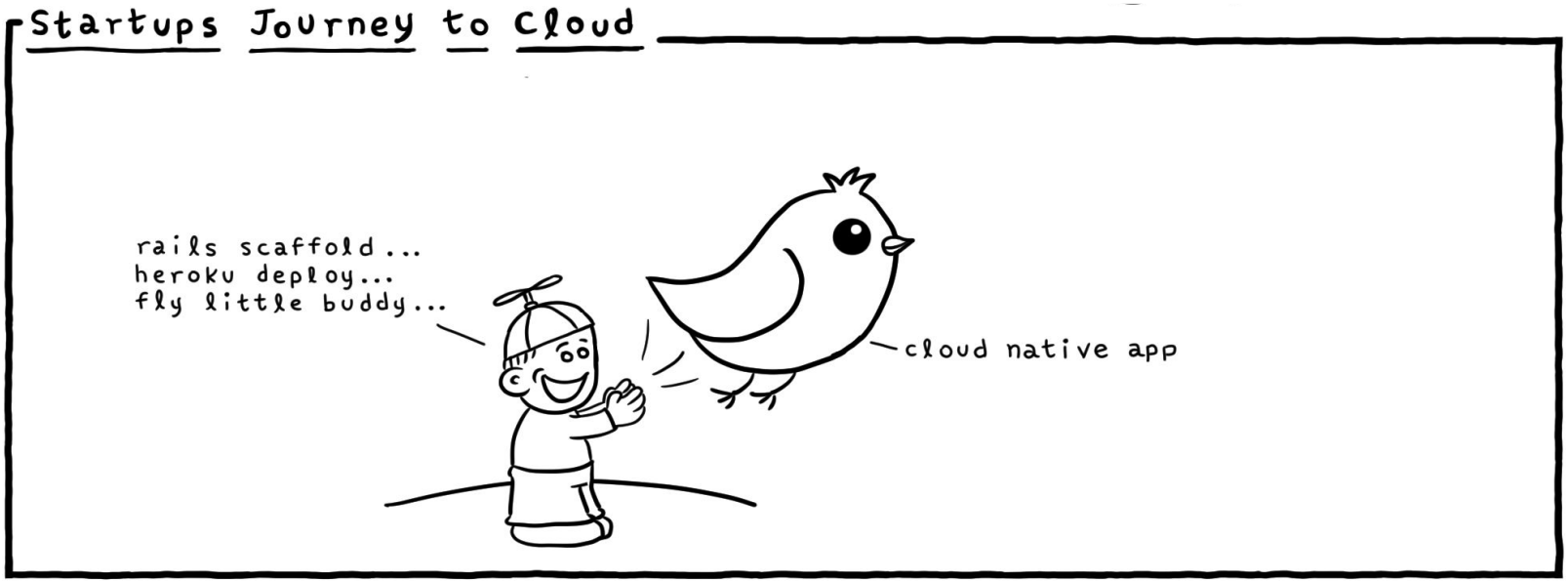
From Bare-Metal Windows to Kubernetes in Two Months

Paul Steele

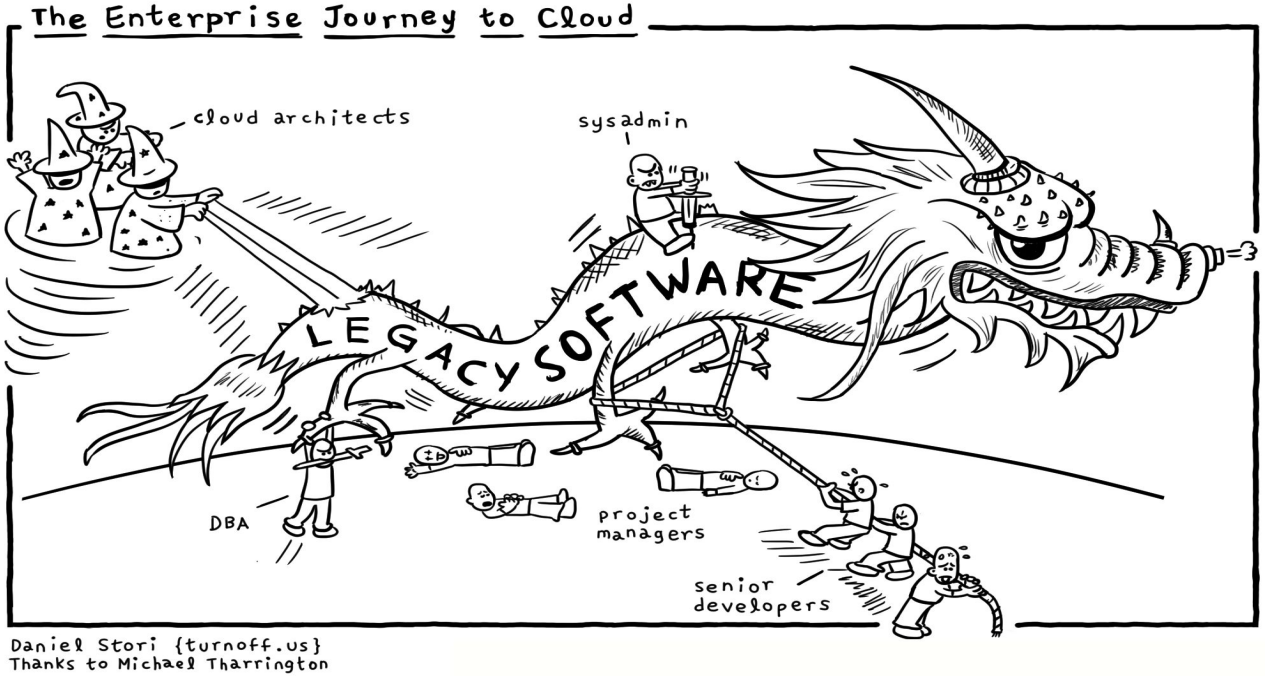
About Me

- Software Engineer at SEP
- Purdue University
- https://blog.paul-steele.com/

Greenfield


Startups Journey to Cloud
rails scaffold...
heroku deploy...
fly little buddy...
cloud native app
Daniel Stori {turnoff.us}
Thanks to Michael Tharrington

Legacy

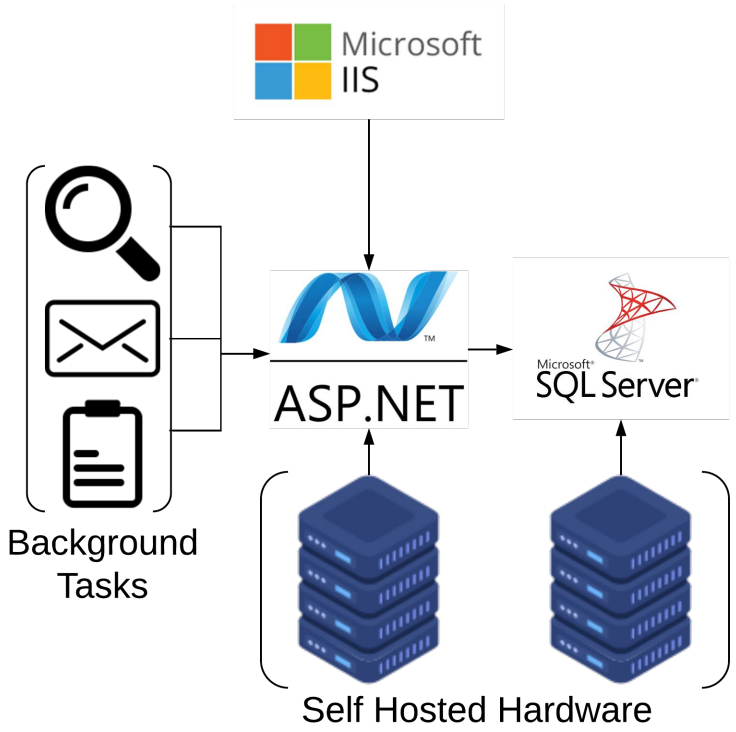

The Enterprise Journey to Cloud
cloud architects
sysadmin
LEGACY SOFTWARE
DBA
project managers
senior developers
Daniel Stori {turnoff.us}
Thanks to Michael Tharrington

Agenda

- Journey of migrating legacy application to the cloud
- Showcase some of the “gotchas” we found

The Application

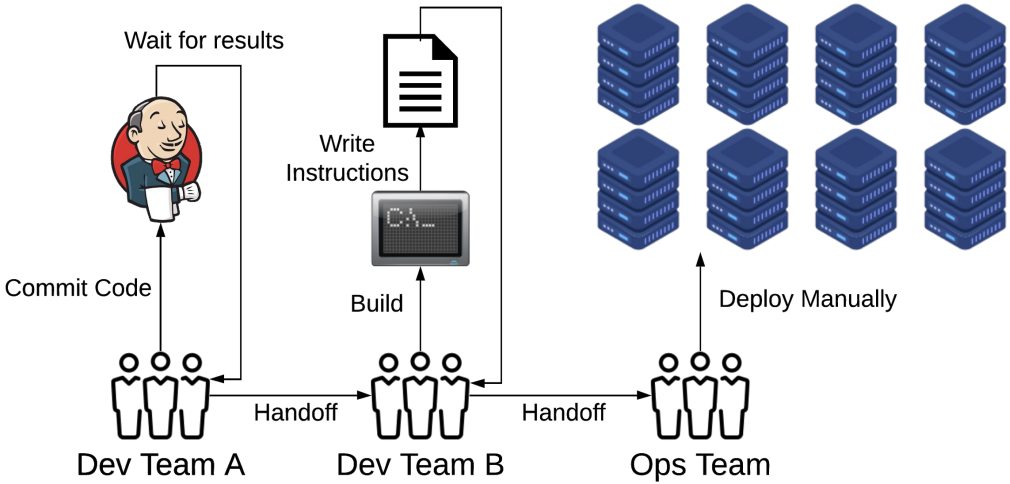
- Incident tracking software
- C# application
  - IIS
  - Self Hosted - Bare Metal
- 10-15 years old
  - Architectural tradeoffs
- Supplemental windows services
  - Background tasks
  - Email
  - Indexing



Feature Development

- Two main teams
  - One in Indianapolis (SEP)
  - One several states away
- Focused solely on Development
- Never had the “budget” for DevOps

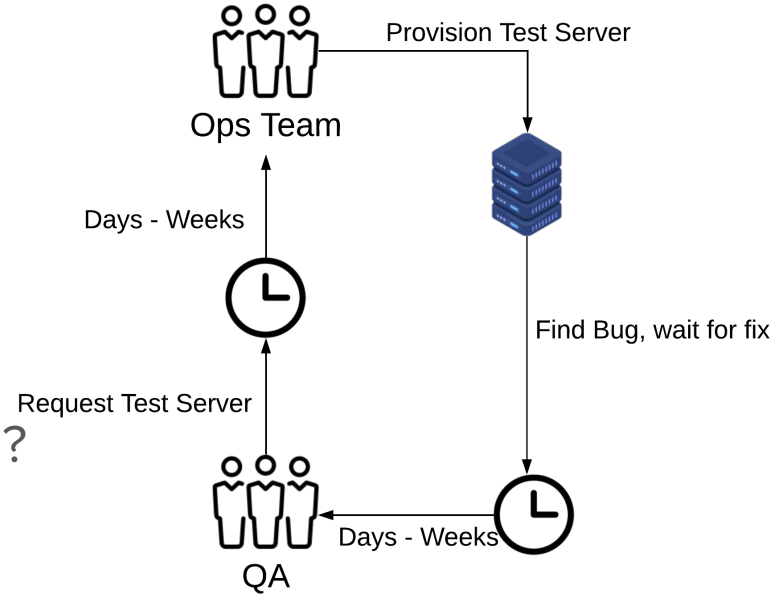
Problems



- Jenkins
  - Old
  - Only accessible to one team
- App hard to deploy
  - Manual file drops
  - Long list of instructions
- No automated database setup
  - No way to create a blank database

What did this cause?

- Uncertainty of state of code
  - Was the master branch passing?
- Slow QA cycles
  - Ask for a server to be setup
    - Days to weeks
  - Check bug
  - Repeat if necessary
- Deployed wrong version to production



Where We Came In

- Budget approved to move to the cloud
  - Azure
  - CI / CD pipeline
  - Deployments
- Wanted to experiment with new technologies
  - Docker
  - Kubernetes
- Team of three
- Two months to pull it off

Goals for the engagement

- Get into the cloud
  - Containerize Application
  - Automated Database Creation
  - Setup Jenkins
  - Push button deployments into test environment
    - Not enough time to get into production
- Everything should be scripted

Containerizing the Application

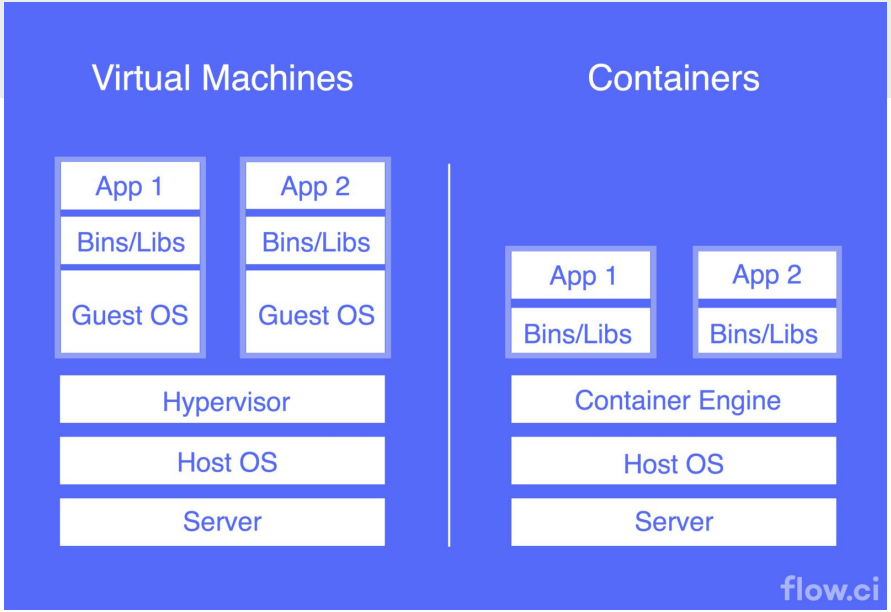
Containerizing The Application

- C# legacy application
  - No .net core
- Can’t use linux containers


WiNDOWS CONTAiNeRS

Containers

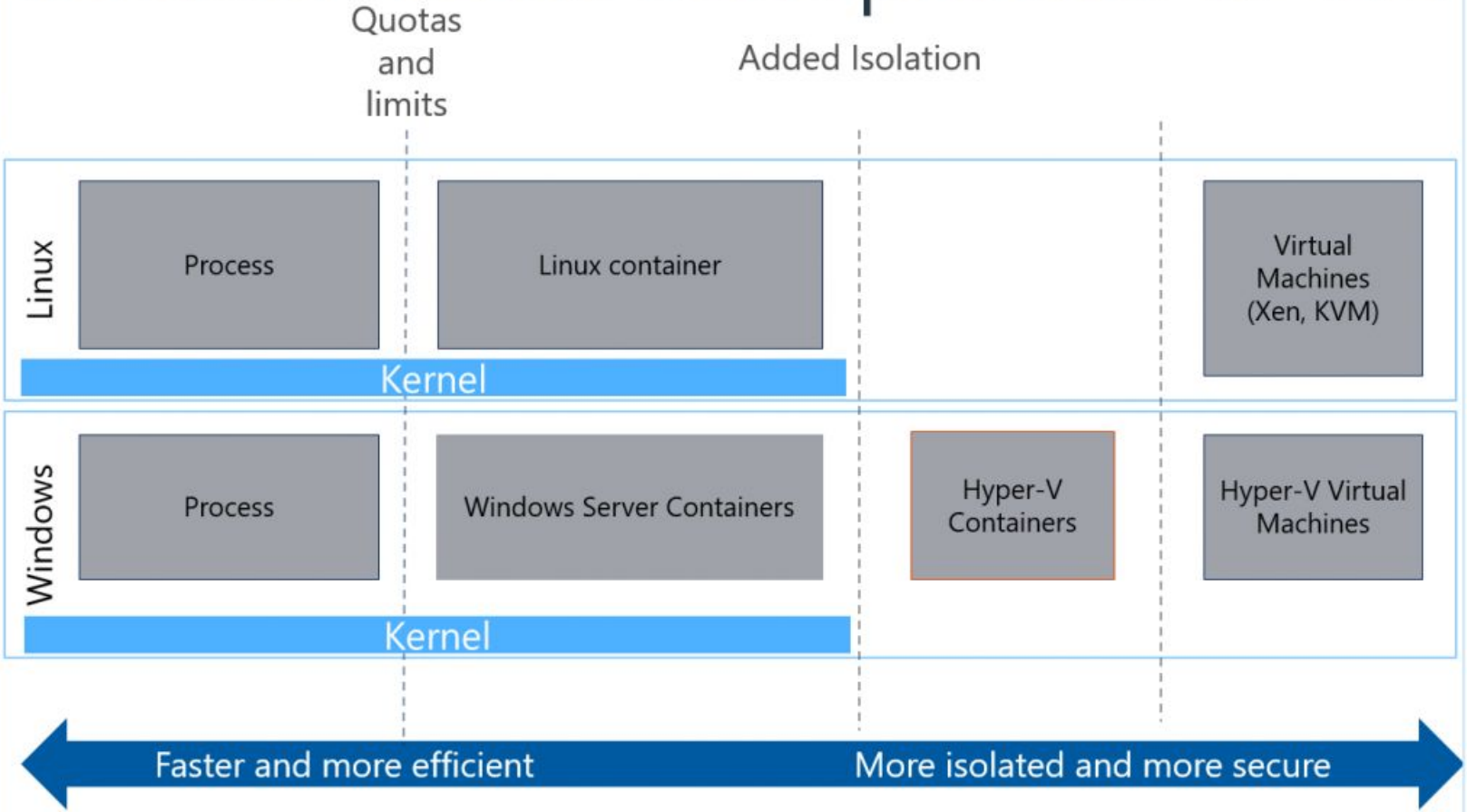
- Share Kernel Space
- Linux
  - Few Compatibility Problems
  - Try to use newer kernel feature
    - Runtime error
- Windows
  - Compatibility Problems
  - Try to use newer kernel feature
    - Startup error



Container OS version	Host OS version							
	Windows Server 2016 Builds: 14393.*	**Windows 10 1609, 1703** Builds: 14393.*, 15063.*	**Windows Server version 1709** Builds 16299.*	**Windows 10 Fall Creators Update** Builds 16299.*	**Windows Server version 1803** Builds 17134.*	**Windows 10 version 1803** Builds 17134.*	**Windows Server 2019** Builds 17763.*	**Windows 10 version 1809** Builds 17763.*
Windows Server 2016 Builds: 14393.*	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1709 Builds 16299.*	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1803 Builds 17134.*	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server 2019 Builds 17763.*	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation

https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Isolation levels from process to VM



https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Windows Isolation Levels

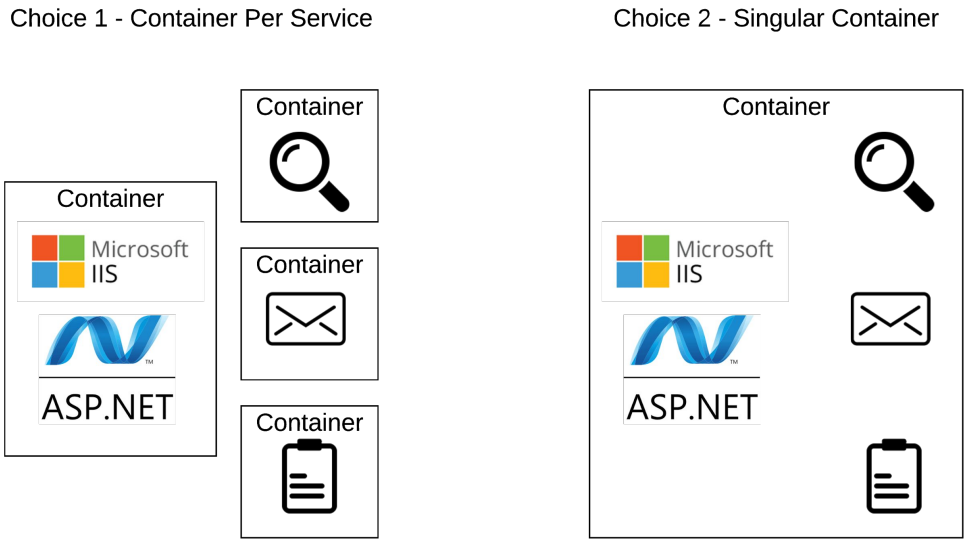
- Process isolation
  - Actual containers
- Hyper-V isolation
  - Pretender containers
- None of this was obvious to us setting out

Container OS version	Host OS version							
	Windows Server 2016 Builds: 14393.*	**Windows 10 1609, 1703** Builds: 14393.*, 15063.*	**Windows Server version 1709** Builds 16299.*	**Windows 10 Fall Creators Update** Builds 16299.*	**Windows Server version 1803** Builds 17134.*	**Windows 10 version 1803** Builds 17134.*	**Windows Server 2019** Builds 17763.*	**Windows 10 version 1809** Builds 17763.*
Windows Server 2016 Builds: 14393.*	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1709 Builds 16299.*	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1803 Builds 17134.*	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server 2019 Builds 17763.*	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation

https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Helper Services

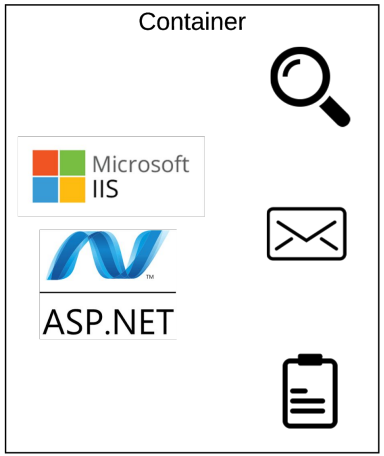
- The container would start up IIS



Helper Services

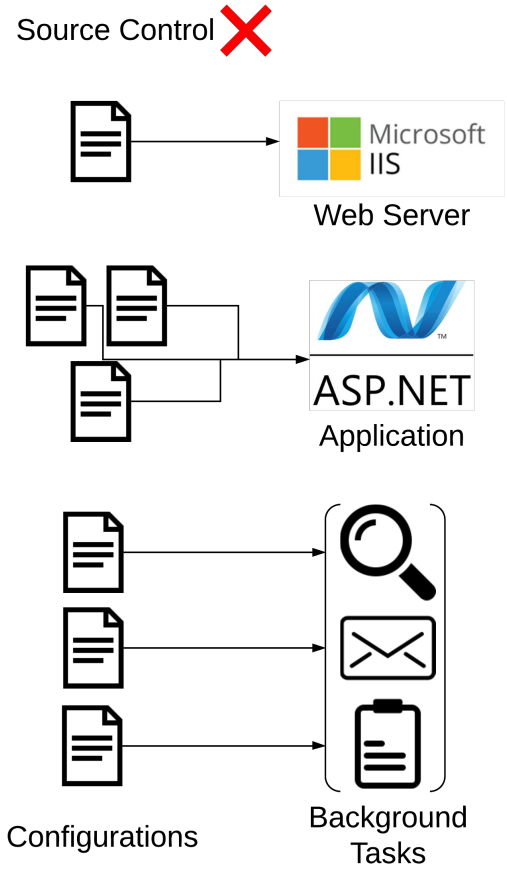
- Chose option 2

Choice 2 - Singular Container



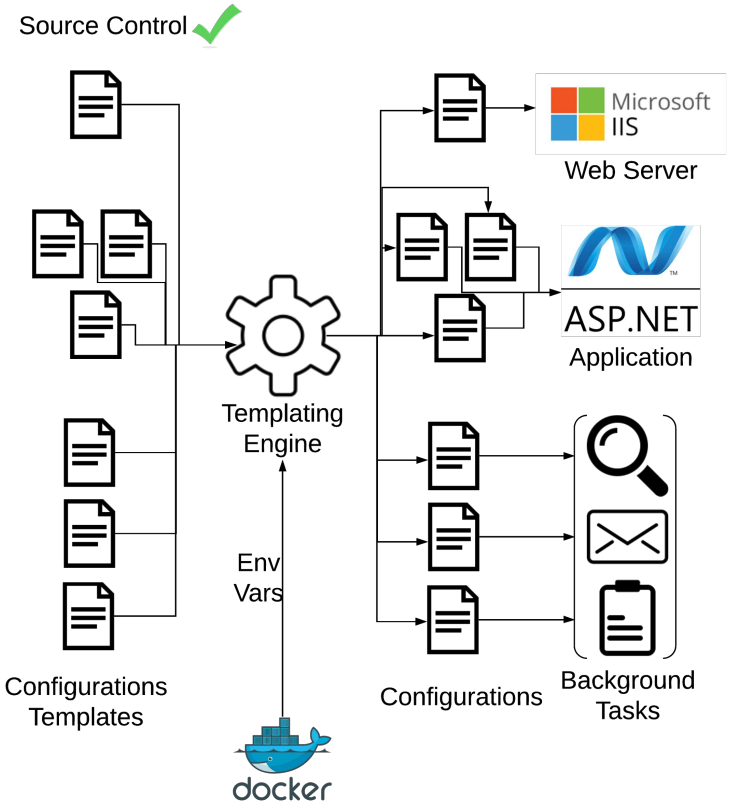
Fundamental Application Problems

- App configurations in various flat files.
  - Not in source control
- IIS configuration
  - Not in source control
- Helper services configuration
  - Not in source control
- Deployment required manually editing these files
  - Highly error prone



Addressing Configuration

- Consolidated configurations
  - Standard location
  - Source control
  - Templated the configurations
  - 95% of configurations standard
  - On container start, fill in templates
- Controlled by environment variables



Goals for the engagement

- Get into the cloud
  - ~~Containerize Application~~
  - Automated Database Creation
  - Setup Jenkins
  - Push button deployments into test environment
- Everything should be scripted

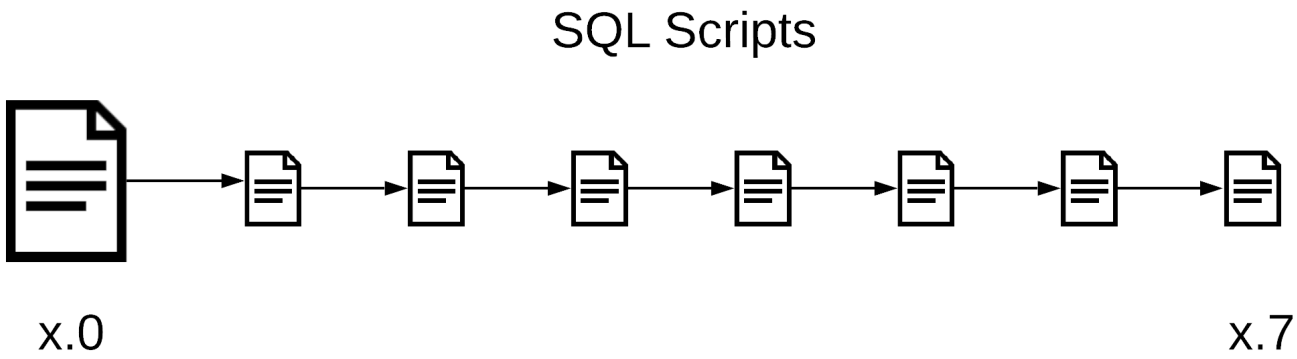
Automated Database Creation

State of the Database

- Microsoft SQL
- No clean setup
- Version x.7 introduced App migrations
  - Years of manual schema updates up to x.7
- Support creation for both
  - Azure SQL
  - Docker Database

A way forward

- Found a series of test scripts
- One script to get to version x
  - 7 scripts for each minor version



The script

- Added test data as it went

```
4
5  def migrate():
6
7      upgrade_to_ver_0()
8      seed_ver_0()
9
10     upgrade_to_ver_1()
11     seed_ver_1()
12
13     upgrade_to_ver_2()
14     seed_ver_2()
15
16     upgrade_to_ver_3()
17     seed_ver_3()
18
19     upgrade_to_ver_4()
20     seed_ver_4()
21
22     upgrade_to_ver_5()
23     seed_ver_5()
24
25     upgrade_to_ver_6()
26     seed_ver_6()
27
28     upgrade_to_ver_7()
29     seed_ver_7()
```

Untangling

- Hopeful we could add some conditionals

```
4
5  def migrate(seed_data):
6
7      upgrade_to_ver_0()
8      if seed_data:
9          seed_ver_0()
10
11     upgrade_to_ver_1()
12     if seed_data:
13         seed_ver_1()
14
15     upgrade_to_ver_2()
16     if seed_data:
17         seed_ver_2()
18
19     upgrade_to_ver_3()
20     if seed_data:
21         seed_ver_3()
22
23     upgrade_to_ver_4()
24     if seed_data:
25         seed_ver_4()
26
27     upgrade_to_ver_5()
28     if seed_data:
29         seed_ver_5()
30
31     upgrade_to_ver_6()
32     if seed_data:
33         seed_ver_6()
34
35     upgrade_to_ver_7()
36     if seed_data:
37         seed_ver_7()
```

Not so Lucky

- Schema scripts relied on some data existing...

```
4
5  def migrate(seed_data):
6
7      upgrade_to_ver_0()
8      if seed_data:
9          seed_ver_0()
10
11     upgrade_to_ver_1()
12     if seed_data:
13         seed_ver_1()
14
15     upgrade_to_ver_2()
16     if seed_data:
17         seed_ver_2()
18
19     upgrade_to_ver_3()
20     if seed_data:
21         seed_ver_3()
22
23     upgrade_to_ver_4()
24     if seed_data:
25         seed_ver_4()
26
27     upgrade_to_ver_5()
28     if seed_data:
29         seed_ver_5()
30
31     upgrade_to_ver_6()
32     if seed_data:
33         seed_ver_6()
34
35     upgrade_to_ver_7()
36     if seed_data:
37         seed_ver_7()
```

Result

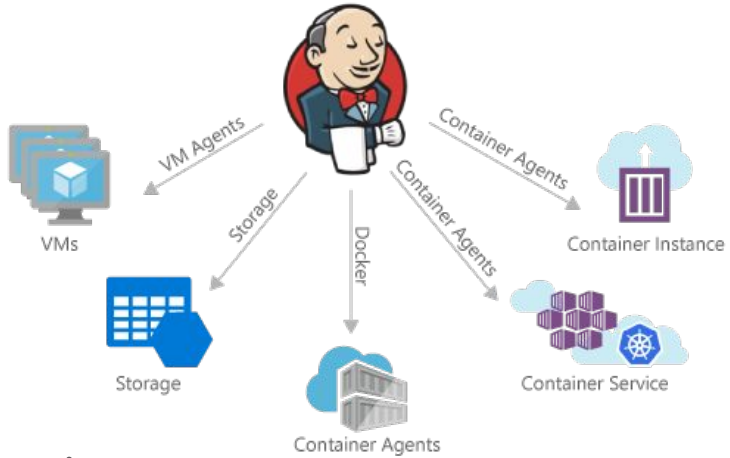
- Fixing took longer than we wanted
- Had the ability to create a database from scratch
  - With / Without test data

Goals for the engagement

- Get into the cloud
  - ~~Containerize Application~~
  - ~~Automated Database Creation~~
  - Setup Jenkins
  - Push button deployments into test environment
- Everything should be scripted

Setup Jenkins

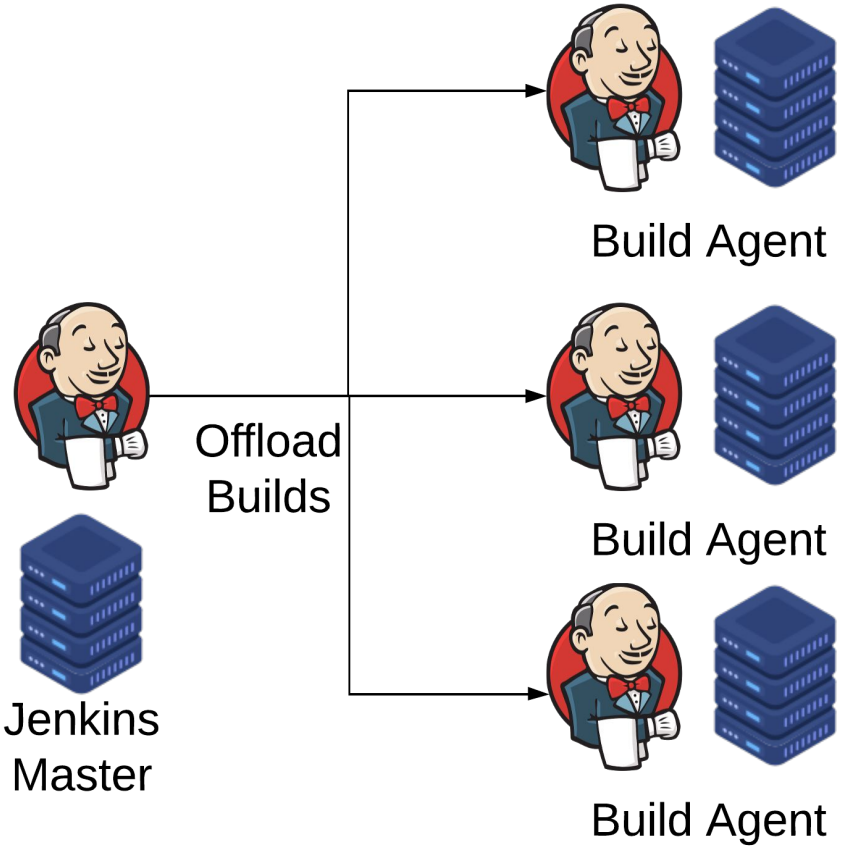
Next Step : Jenkins



- Solution template
  - Azure Marketplace
  - All the basics to jump start a jenkins instance
- Utilized packer to create master image, and build agent images
- Ondemand Build Agents
- Artifact Storage
  - Could use to deploy past testing

https://azuremarketplace.microsoft.com/en-us/marketplace/apps/azure-oss.jenkins

Structure of Jenkins



Build Agents

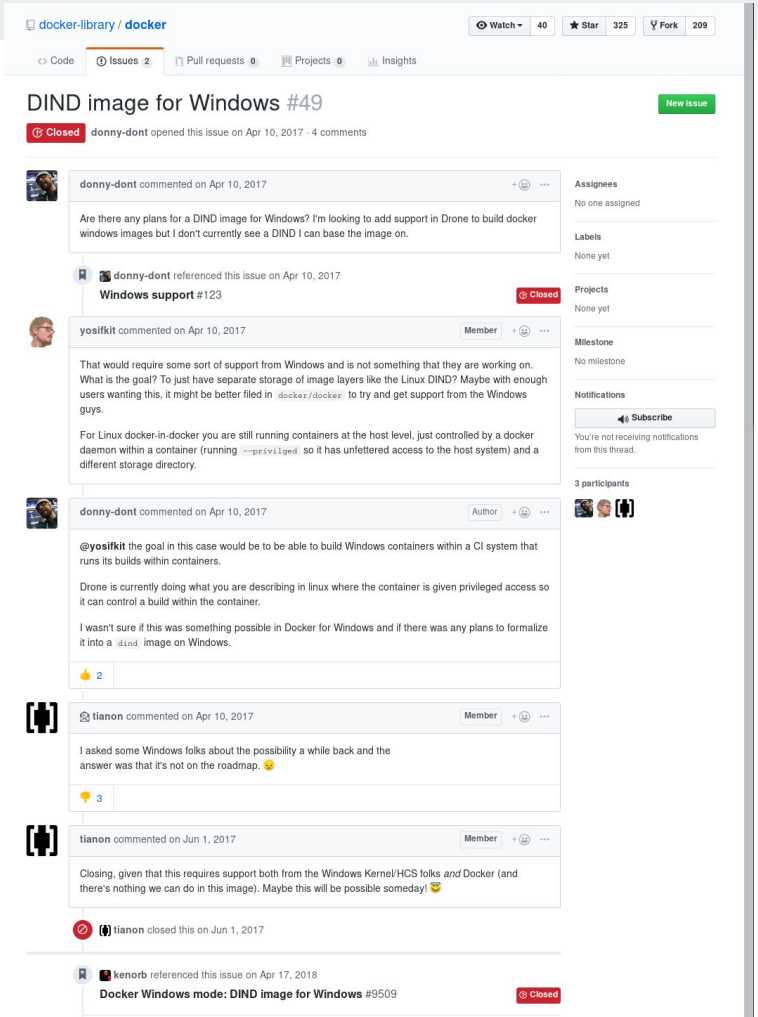
- Virtual Machines
  - Finer control of agents themselves
    - Cpu / memory
    - More to manage
  - Slower to spin up
    - Tend to reuse
  - Expensive
- Containers
  - Less control of agents
    - Less to manager
  - Faster to spin up
  - Take advantage of existing infrastructure
    - Kubernetes

Container build agents sound like the clear choice right?

We Chose Virtual Machines

Why

- Build agents need to build our containers
- Need Docker in Docker
  - Not supported in Windows



docker-library / docker

Watch 40 · Star 325 · Fork 209

Code · Issues 2 · Pull requests 0 · Projects 0 · Insights

DIND image for Windows #49

New issue

Closed donny-dont opened this issue on Apr 10, 2017 · 4 comments

donny-dont commented on Apr 10, 2017

Are there any plans for a DIND image for Windows? I'm looking to add support in Drone to build docker windows images but I don't currently see a DIND I can base the image on.

donny-dont referenced this issue on Apr 10, 2017

Windows support #123 Closed

yosifkit commented on Apr 10, 2017 Member

That would require some sort of support from Windows and is not something that they are working on. What is the goal? To just have separate storage of image layers like the Linux DIND? Maybe with enough users wanting this, it might be better filed in `docker/docker` to try and get support from the Windows guys.

For Linux docker-in-docker you are still running containers at the host level, just controlled by a docker daemon within a container (running `--privilged` so it has unfettered access to the host system) and a different storage directory.

donny-dont commented on Apr 10, 2017 Author

@yosifkit the goal in this case would be to be able to build Windows containers within a CI system that runs its builds within containers.

Drone is currently doing what you are describing in linux where the container is given privileged access so it can control a build within the container.

I wasn't sure if this was something possible in Docker for Windows and if there was any plans to formalize it into a `dind` image on Windows.

2

tianon commented on Apr 10, 2017 Member

I asked some Windows folks about the possibility a while back and the answer was that it's not on the roadmap.

3

tianon commented on Jun 1, 2017 Member

Closing, given that this requires support both from the Windows Kernel/HCS folks *and* Docker (and there's nothing we can do in this image). Maybe this will be possible someday!

tianon closed this on Jun 1, 2017

kenorb referenced this issue on Apr 17, 2018

Docker Windows mode: DIND image for Windows #9509 Closed

Assignees
No one assigned

Labels
None yet

Projects
None yet

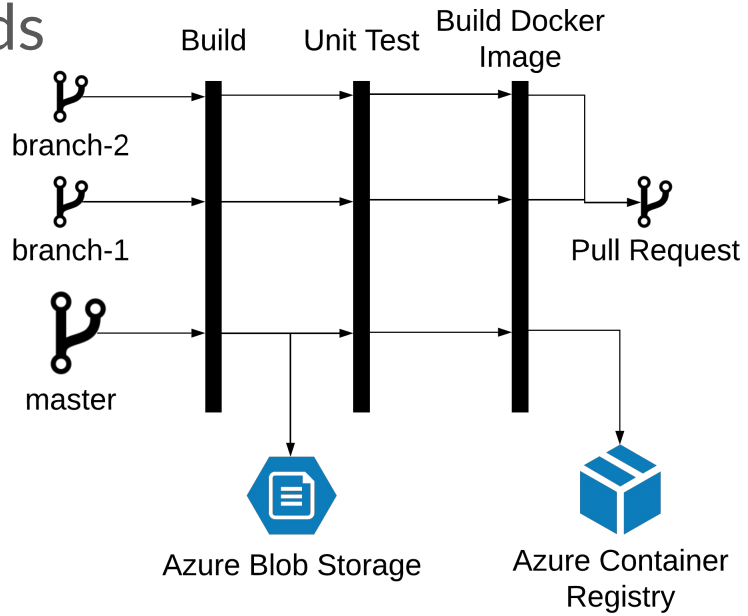
Milestone
No milestone

Notifications
Subscribe
You're not receiving notifications from this thread.

3 participants

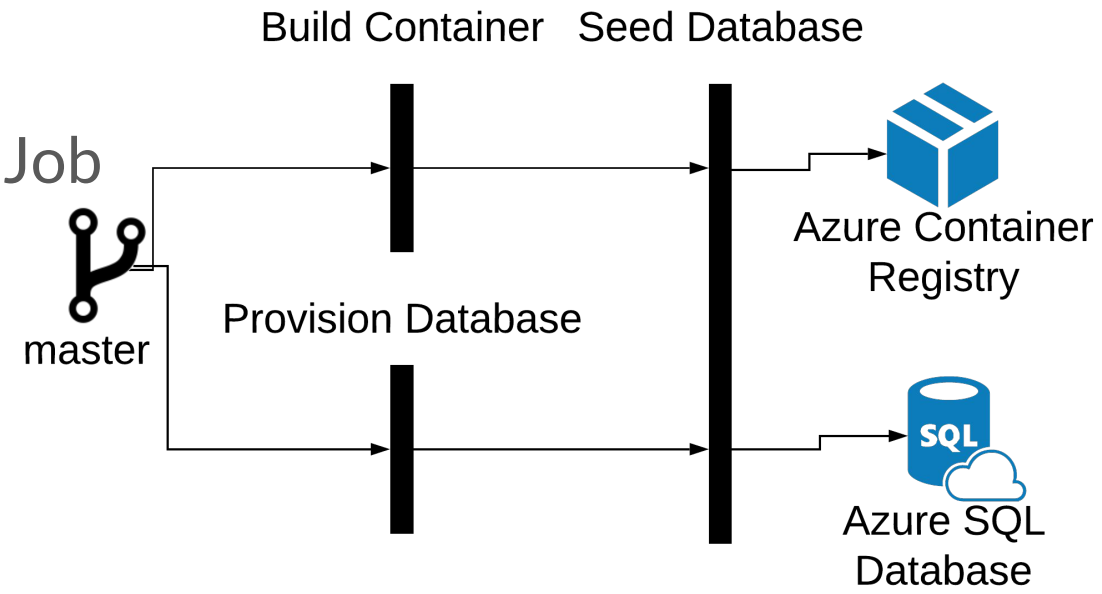
Jenkins Overview

- Multibranch pipeline for builds
  - Didn’t have CI builds for branches before
  - Didn’t build application in dockerfile
    - Archive to azure blob storage



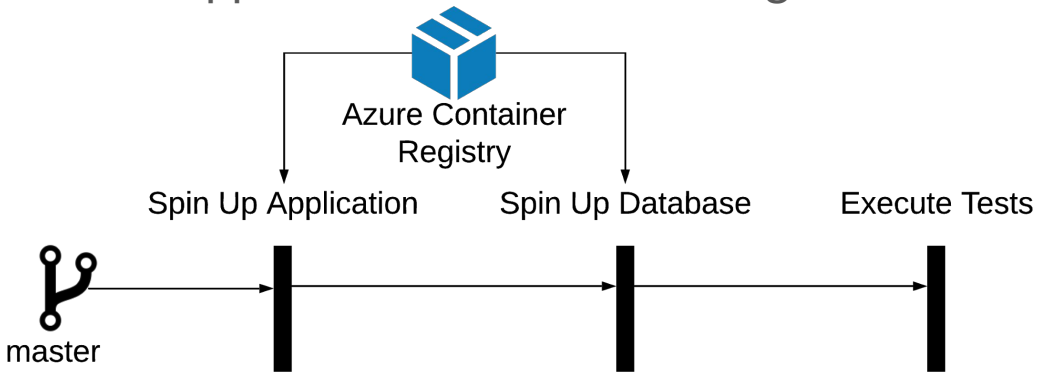
Database Creation

- Database Creation Job
  - System Testing
  - QA environments



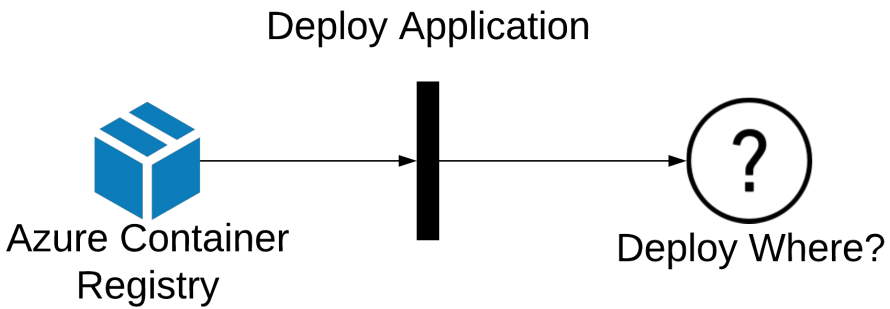
System Tests

- System Test Job
  - Ran Integration tests for master branch
  - Challenge to modify the tests to work in Jenkins
    - Ran the app container in on the vm agents



Deployment Job

- Application Deployment Job
  - Deploy container
  - Deploy Where?

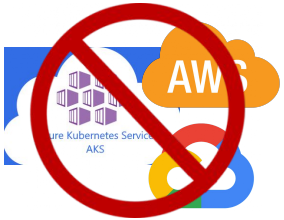


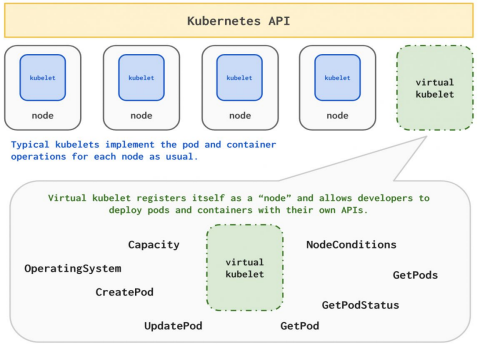
Goals for the engagement

- Get into the cloud
  - ~~Containerize Application~~
  - ~~Automated Database Creation~~
  - ~~Setup Jenkins~~
  - Push button deployments into test environment
- Everything should be scripted

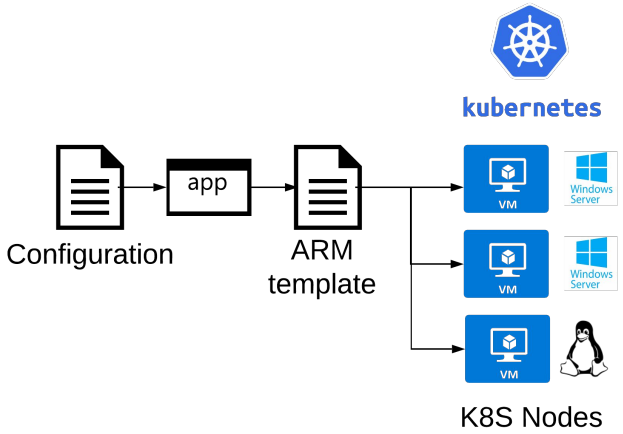
Push Button Deployment

Enter Kubernetes



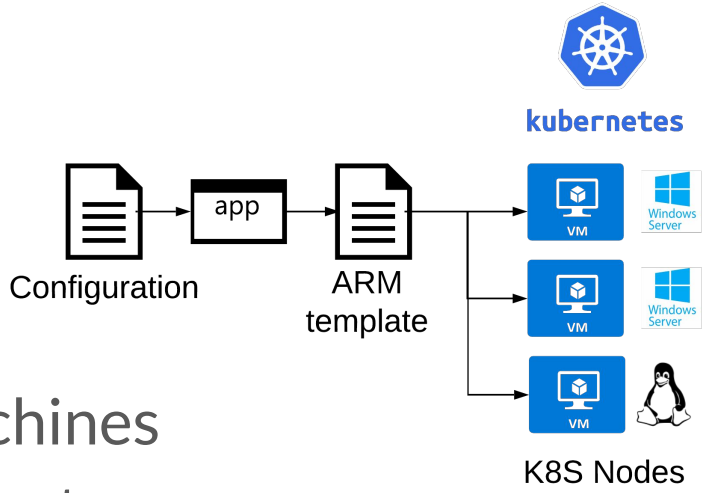


- Azure? Use AKS!
  - Windows containers not natively supported
  - Better than AWS, or Google Cloud
- Options?
  - Virtual Kubelet
  - acs-engine



https://github.com/virtual-kubelet/virtual-kubelet#how-it-works

ACS Engine

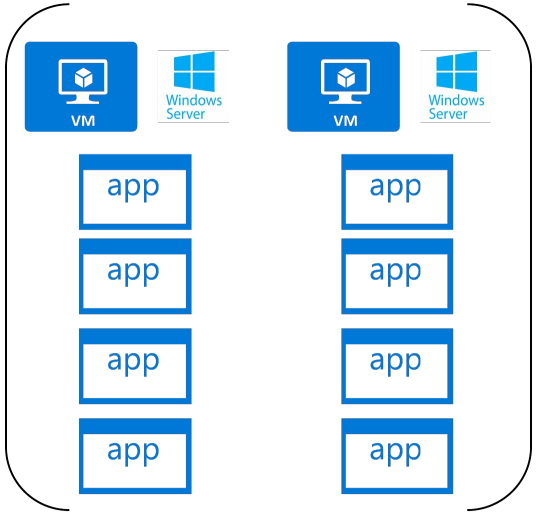


- Generated Solution Templates
- “Mimic’d” ACS with virtual machines
- Allowed creating Kubernetes Instances
  - Supported swarm
- Supported Hybrid Clusters
- Has since been deprecated for aks-engine
- Config files for ACS stored in source control

Kubernetes

- Master Node
  - Linux
  - Ran Nginx Ingress Controller + Certmanager
- Other Nodes
  - Windows
  - With our VM choice could fit 4 Applications per node
    - ACS engine provided node auto scaling

Master Node



App Nodes

Goals for the engagement

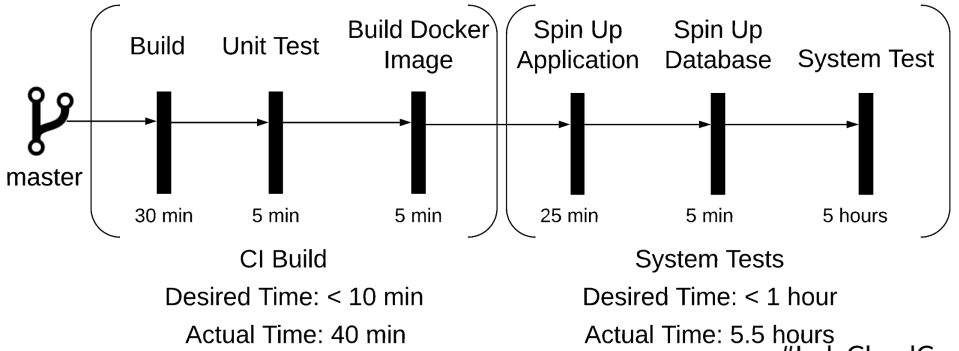
- Get into the cloud
  - ~~Containerize Application~~
  - ~~Automated Database Creation~~
  - ~~Setup Jenkins~~
  - ~~Push button deployments into test environment~~
- ~~Everything should be scripted~~

So We're Done!!!

Right?

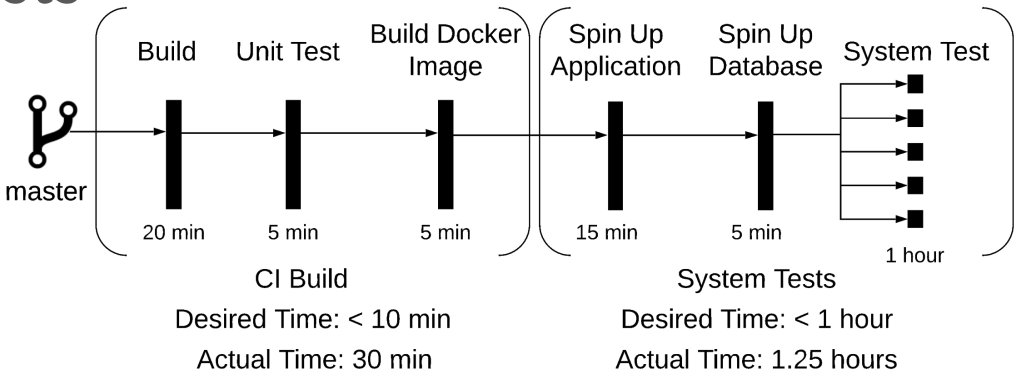
Aren't we Done?

- Jenkins performance was pitiful
  - Large base images
    - 10 minutes to pull base layer
    - Slow to build the container
  - System Tests
    - Inconsistent Failures
    - Slow

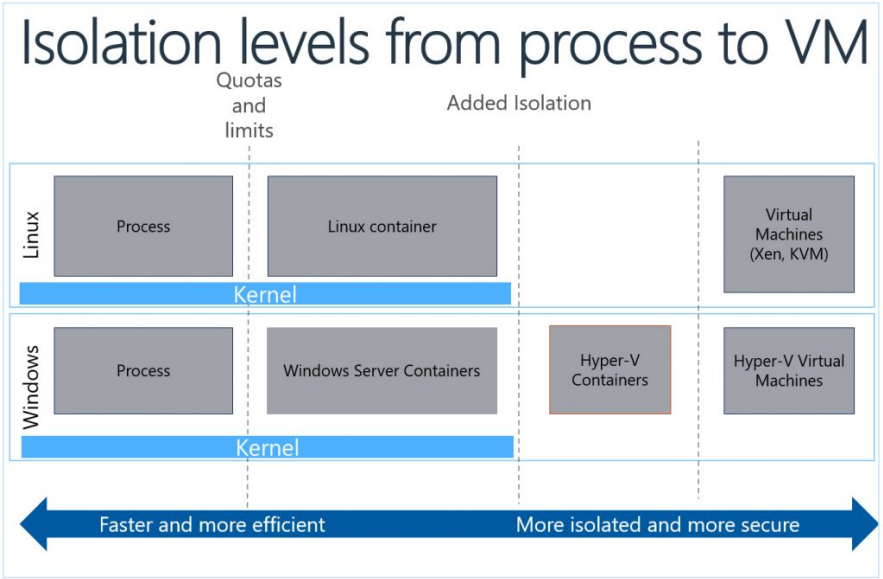


Quick Fixes

- Pull base layer during image creation
  - Adds extra time to image creation
    - Better than every build
- Parallelize System Tests
  - From 1 node to 5
    - Budget constraints



Larger Problem



ot process isolation

https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Container OS version	Host OS version							
	Windows Server 2016 Builds: 14393.*	**Windows 10 1609, 1703** Builds: 14393.*, 15063.*	**Windows Server version 1709** Builds 16299.*	**Windows 10 Fall Creators Update** Builds 16299.*	**Windows Server version 1803** Builds 17134.*	**Windows 10 version 1803** Builds 17134.*	**Windows Server 2019** Builds 17763.*	**Windows 10 version 1809** Builds 17763.*
Windows Server 2016 Builds: 14393.*	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1709 Builds 16299.*	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1803 Builds 17134.*	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server 2019 Builds 17763.*	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation

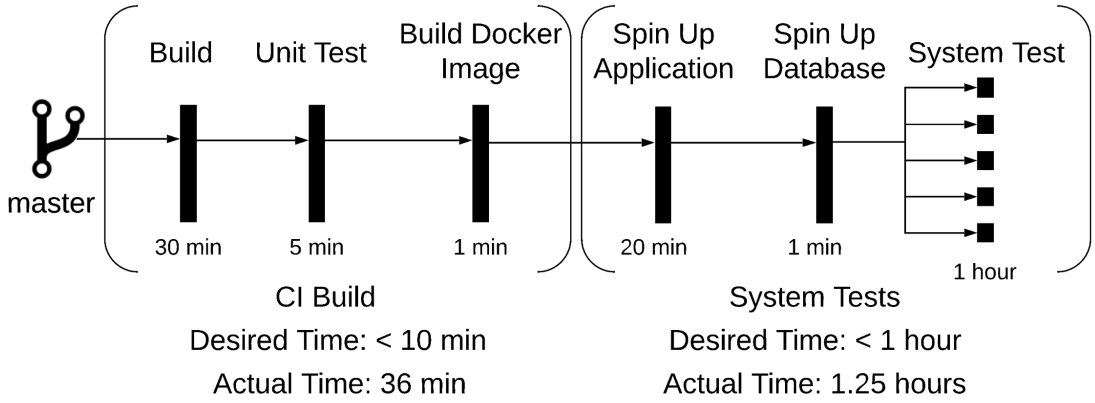
https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Container OS version	Host OS version							
	Windows Server 2016 Builds: 14393.*	**Windows 10 1609, 1703** Builds: 14393.*, 15063.*	**Windows Server version 1709** Builds 16299.*	**Windows 10 Fall Creators Update** Builds 16299.*	**Windows Server version 1803** Builds 17134.*	**Windows 10 version 1803** Builds 17134.*	**Windows Server 2019** Builds 17763.*	**Windows 10 version 1809** Builds 17763.*
Windows Server 2016 Builds: 14393.*	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1709 Builds 16299.*	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1803 Builds 17134.*	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server 2019 Builds 17763.*	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation

https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

What could go wrong?

- VM agents started taking twice as long to spin up
  - 20 minutes at worst case
- Container start time did improve however



System Tests

- System Tests were still flakey
- Narrowed it down to Time skew
  - 30 minute offset inside the container

The solution

- Use a different base image,
  - Upgrade to the latest and greatest
  - Continue to use process isolation

Container OS version	Host OS version							
	Windows Server 2016 Builds: 14393.*	**Windows 10 1609, 1703** Builds: 14393.*, 15063.*	**Windows Server version 1709** Builds 16299.*	**Windows 10 Fall Creators Update** Builds 16299.*	**Windows Server version 1803** Builds 17134.*	**Windows 10 version 1803** Builds 17134.*	**Windows Server 2019** Builds 17763.*	**Windows 10 version 1809** Builds 17763.*
Windows Server 2016 Builds: 14393.*	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1709 Builds 16299.*	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1803 Builds 17134.*	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server 2019 Builds 17763.*	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation

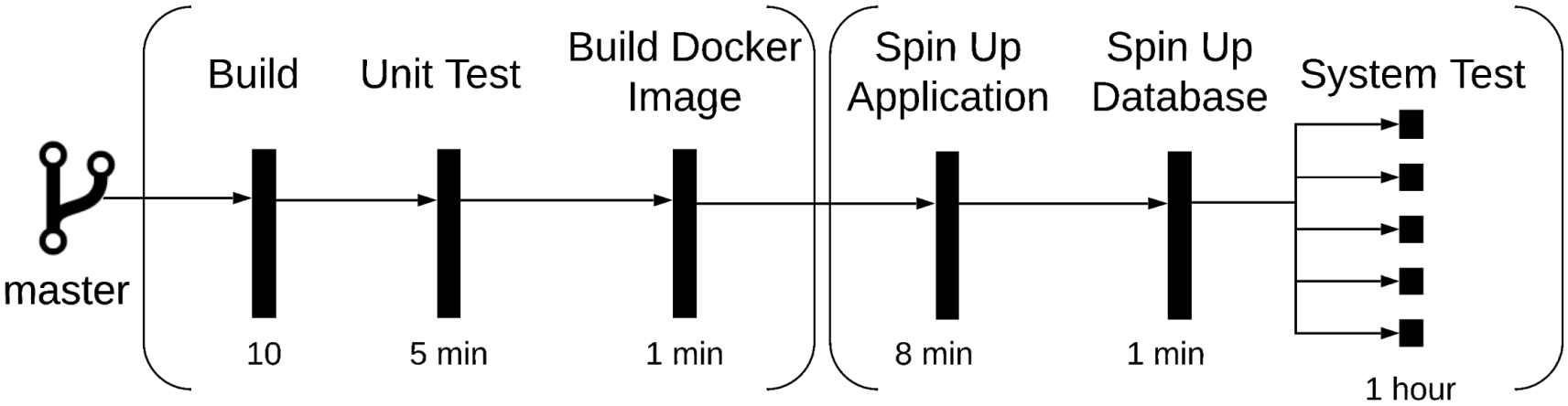
https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Container OS version	Host OS version							
	Windows Server 2016 Builds: 14393.*	**Windows 10 1609, 1703** Builds: 14393.*, 15063.*	**Windows Server version 1709** Builds 16299.*	**Windows 10 Fall Creators Update** Builds 16299.*	**Windows Server version 1803** Builds 17134.*	**Windows 10 version 1803** Builds 17134.*	**Windows Server 2019** Builds 17763.*	**Windows 10 version 1809** Builds 17763.*
Windows Server 2016 Builds: 14393.*	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1709 Builds 16299.*	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server version 1803 Builds 17134.*	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation	Supports Only `hyperv` isolation
Windows Server 2019 Builds 17763.*	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported	Supports `process` or `hyperv` isolation	Supports Only `hyperv` isolation

https://blogs.msdn.microsoft.com/azureservicefabric/2016/04/25/orchestrating-containers-with-service-fabric/

Keep VM Agents around

- Half of the time for CI builds was waiting for Vms
- Keep them around for 1 hour



CI Build

Desired Time: < 10 min

Actual Time: 16 min

Actual Time - Agent Spin up Time: 8 min

System Tests

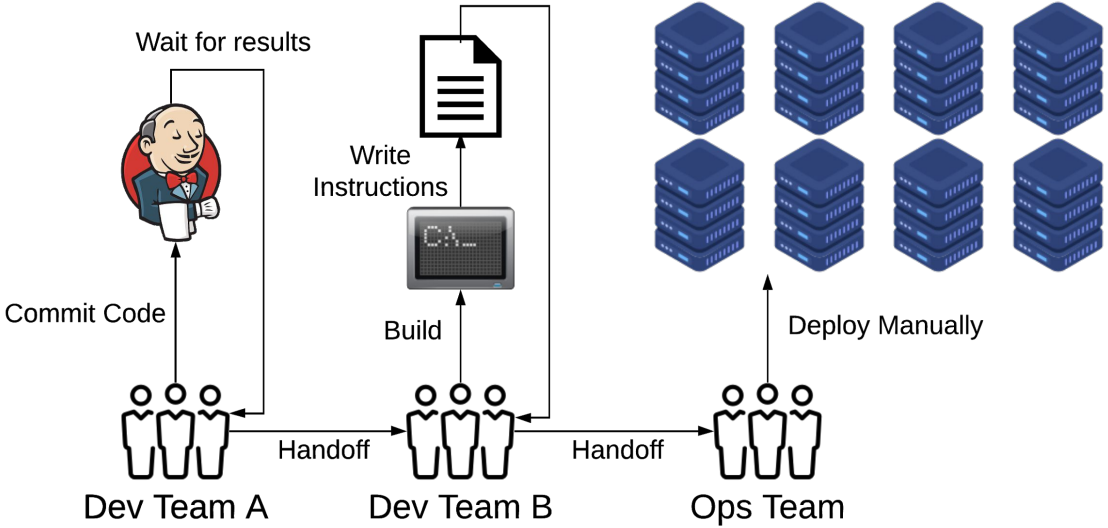
Desired Time: < 1 hour

Actual Time: 1.1 hours

Results after Two months

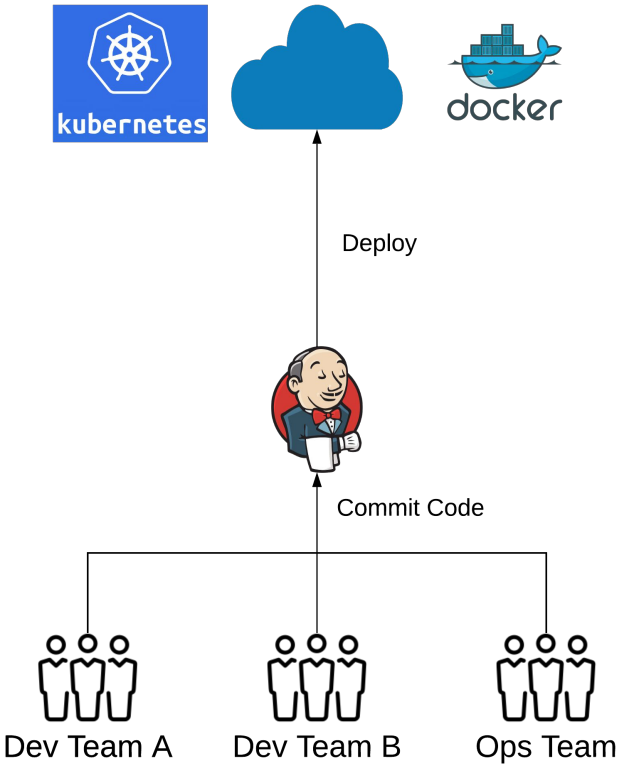
- Successful System Tests
- VM agents
  - Spun in 7 minutes
  - Persist between builds
- CI builds
  - Best Case < 10 minutes
- Push Button Deployment to Kubernetes

Before



Months to deploy

After



Hours to Deploy

Key Takeaways

- Tooling for windows containers is lacking
  - Until 1803, not worth trying
    - Slow
    - Buggy
- If we didn’t script all of this, never would have gotten done
  - Took a little longer upfront
  - All those vm changes, required full rebuild
- Cloud costs need to be monitored

Questions?

Thanks!

- Slides can be found: https://info.sep.com/2019indycloudconf
- SEP blog: https://www.sep.com/sep-blog/2019/04/11/migrating-a-legacy-asp-net-application-to-azure/