Building FAST Data Solutions with DC/OS on Azure

Rob Bagby

Sr. Software Development Engineer

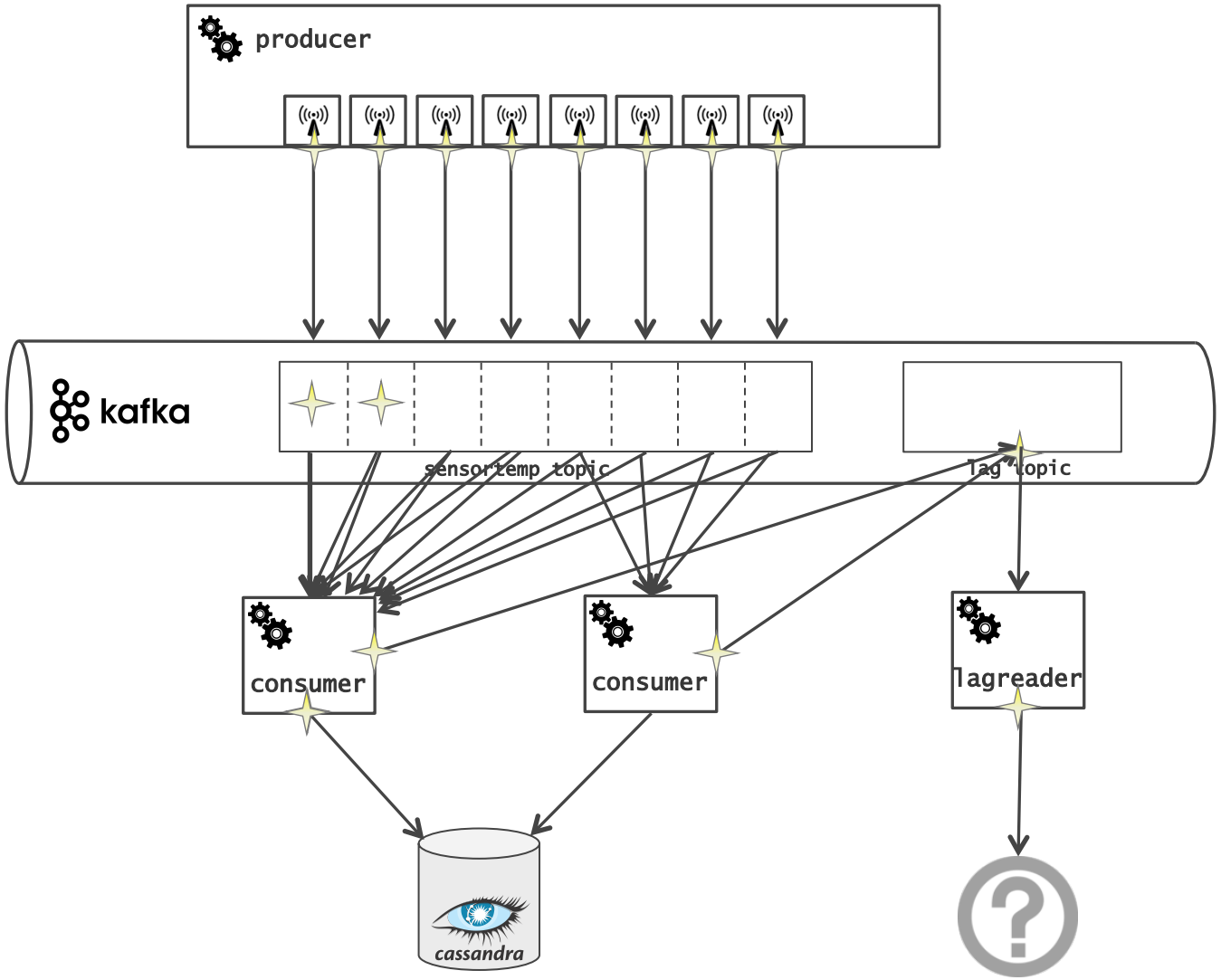
rob.bagby@microsoft.com

FAST Data and the SMACK stack

- FAST Data
  - IoT-type solutions
  - Speed of response time is crucial
- SMACK Stack
  - Spark
  - Mesos
  - Akka
  - Cassandra
  - Kafka

This Session is DEMO-Driven

Application Overview



We will illustrate enabling:

<u>What</u>

1. Development
2. Running at scale
   - Big Data Solutions
   - Data Persistence
3. Managing at scale
   - Autoscaling
   - Workflows

<u>How</u>

Containers

Orchestrator – DC/OS

DC/OS

Portworx

Workflow Solution

VAMP

Challenges containers address

- Running Cassandra / Kafka for Development
- Dependency issues
- Enabling application density

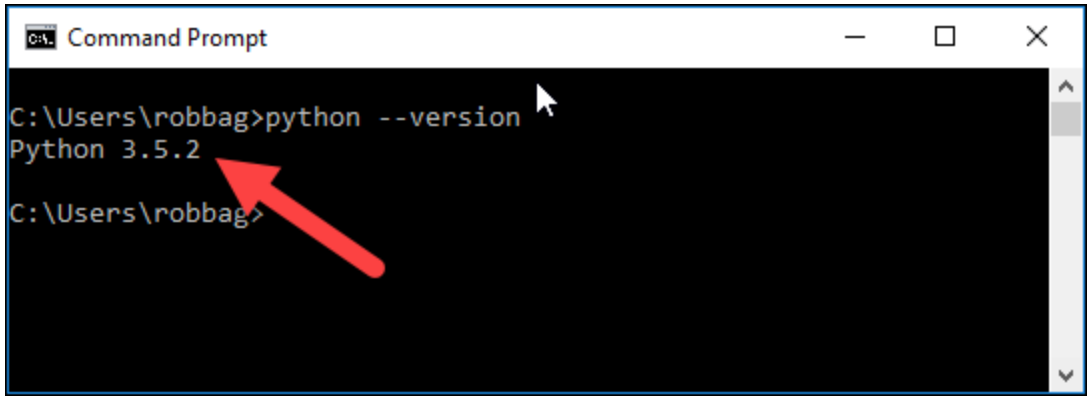
Running Cassandra / Kafka for Development

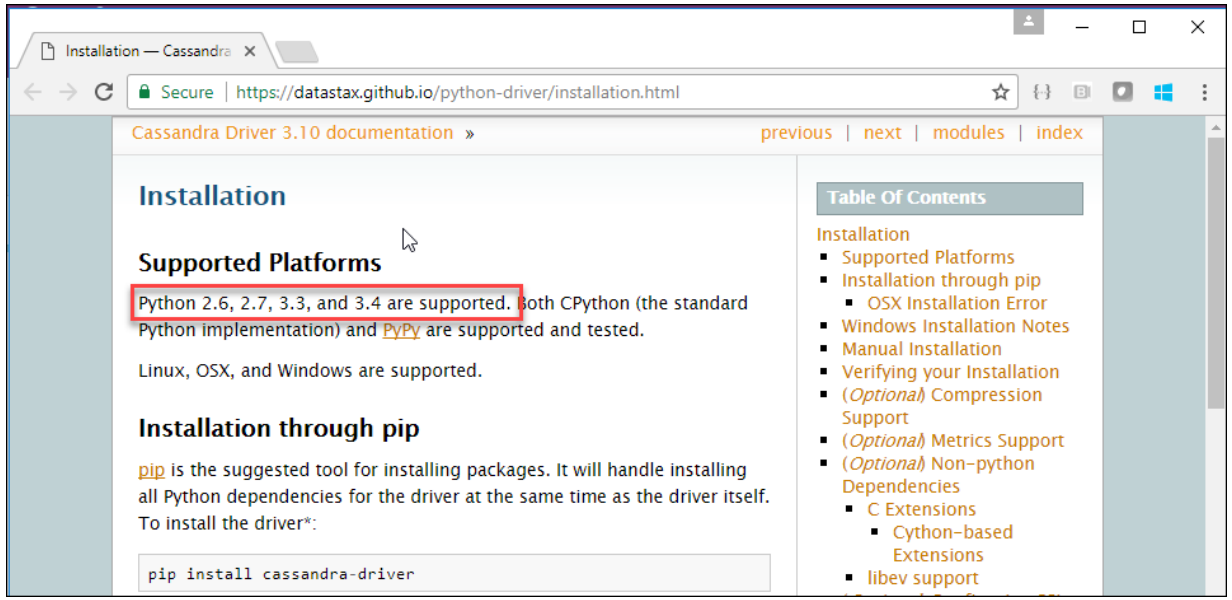
- Traditional Options
  - Install locally – very difficult
  - Shared instances
    - Step on one another
    - Not portable
- Containerized

```
docker run -d --name localcassandra -p 9042:9042 --network=sensor-network
  -v  C:/data:/var/lib/cassandra cassandra:3.10
```

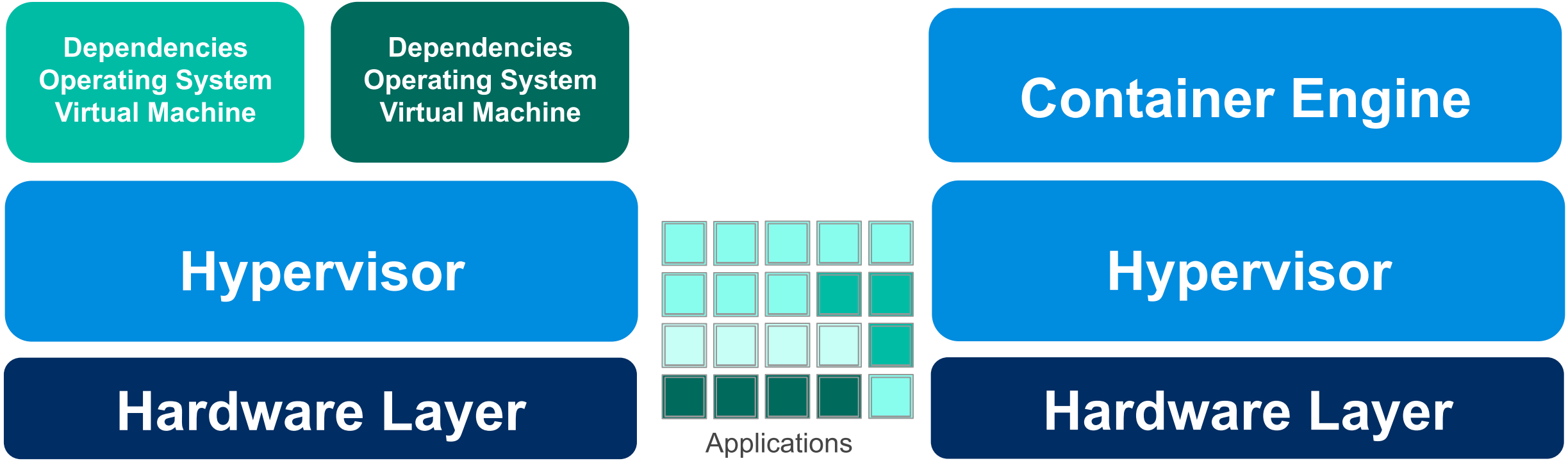
```
docker run -d --name kafka -p 2181:2181 -p 9092:9092 --network=sensor-network
  -env ADVERTISED_HOST=172.30.0.1 --env ADVERTISED_PORT=9092  spotify/kafka
```

Dependency Issues





Dependency encapsulation enables density



Demo – Developing Locally

Rob Bagby

We will illustrate enabling:

What

1. Development
2. Running at scale
   - Big Data Solutions
   - Data Persistence
3. Managing at scale
   - Autoscaling
   - Workflows

How

Containers

Orchestrator

DC/OS

Portworx

Workflow Solution

VAMP

Challenges orchestrators address

- Treat multiple hosts as a single unit
- Determine where containers are started
- Monitor health of containers / applications
- Orchestrate application density
- Allow you to scale services

DC/OS Superpower

(vetted) Frameworks / Services

Mesos Frameworks

- Applications that run on Mesos
- Distributed applications
  - Controller – called the “Scheduler”
  - Workers – called “Executors”
- Frameworks are “Cluster Aware”
  - Specific needs / requirements of the application
  - Cluster resources
  - External triggers
  - …

Example – Cassandra Framework

	Cassandra on DC/OS	Cassandra Bare-Metal/ VM
Installation	Automated	Manual
Dynamic Resource Allocation & Resizing	Yes	VMs Only (Complex)
Node Scaling	Automated	Manual
Multi Datacenter Replication	Simple	Complex
Readiness Checks	Yes	No
Management	Simple & Integrated	Difficult & Isolated
HA Node Replace/Restart	Automated	Manual
Troubleshooting	Simple	Difficult

Custom DC/OS clusters on Azure: acs-engine

Azure/acs-engine: Azur

GitHub, Inc. [US] github.com/Azure/acs-engine

This repository Search Pull requests Issues Marketplace Gist

Azure / acs-engine

Unwatch 89 Star 358 Fork 198

Code Issues 123 Pull requests 26 Projects 1 Wiki Insights

Azure Container Service Engine - a place for community to collaborate and build the best open Docker container infrastructure for Azure.

kubernetes dcos mesos docker swarm swarmmode orchestration containers azure

873 commits 9 branches 5 releases 56 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

seanknox committed on GitHub docs(github): include acs-engine version in issue template (#943) Latest commit 52eff7c 21 minutes ago

.github	docs(github): include acs-engine version in issue template (#943)	21 minutes ago
cmd	delay wrapping of parameters (#861)	17 days ago
docs	update kubernetes doc (#860)	5 days ago
examples	added regression test categories (#919)	4 days ago
parts	Enable cloudprovider rate limit / backoff features (#892)	4 days ago
pkg	populate default value in unmarshal (#939)	2 days ago
scripts	ref(*): remove dead code	17 days ago
test	CI improvements (#932)	3 days ago
vendor	add validate tag for the purpose of required field (#886)	3 days ago
.dockerignore	fix(dockerfile): pin version more correctly	19 days ago

https://github.com/issues

We will illustrate enabling:

What

1. Development
2. Running at scale
   - Big Data Solutions
   - Data Persistence
3. Managing at scale
   - Autoscaling
   - Workflows

How

Containers

Orchestrator

DC/OS

Portworx

Workflow Solution

VAMP

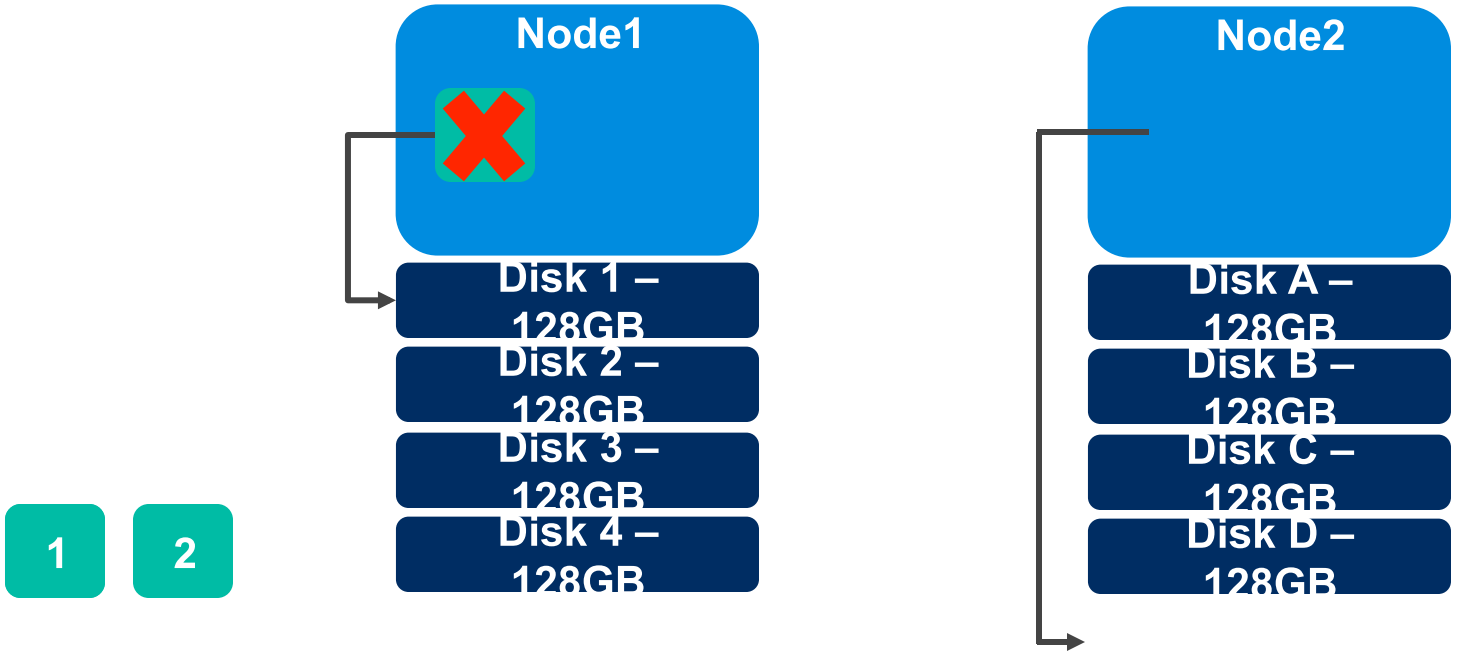
Options for container persistence - Azure

- Azure Files
- VMs / Ephemeral Disks
- Attached / Managed Disks
- Pooled Storage
  - GlusterFS
  - Portworx
  - …

Challenges with Attached / Managed Disks

1. Container Rescheduling

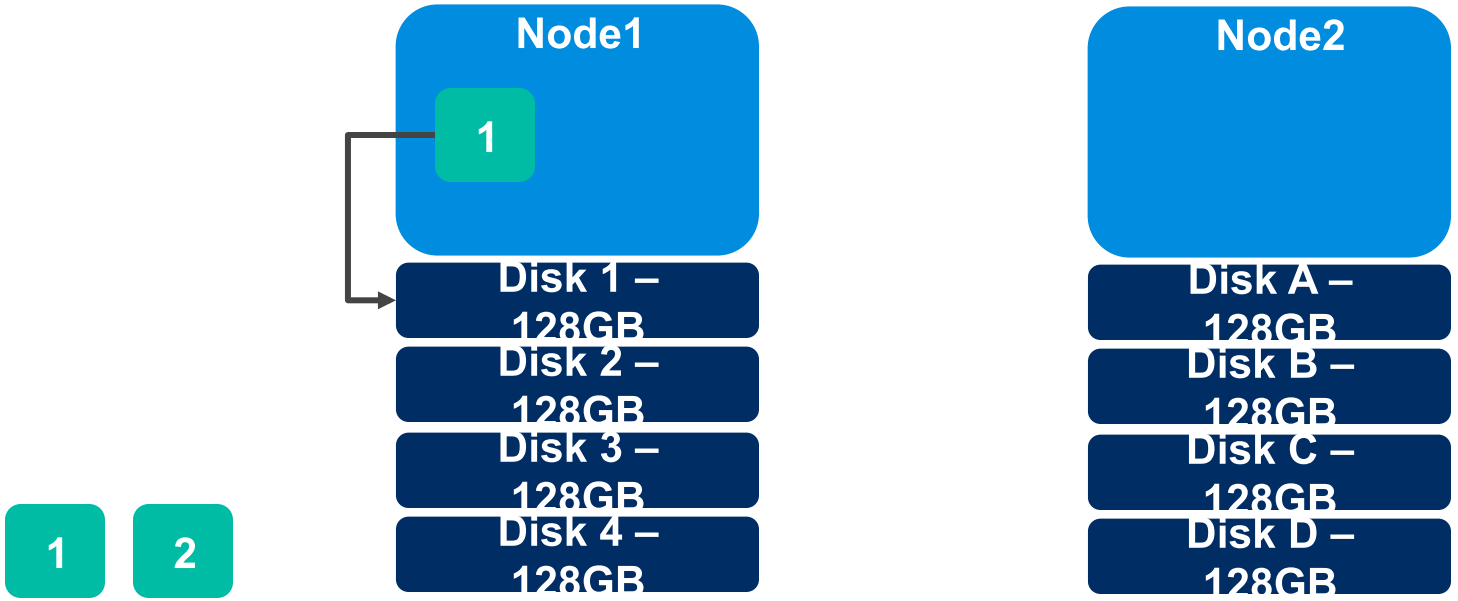
- disk has to move



Challenges with Attached / Managed Disks

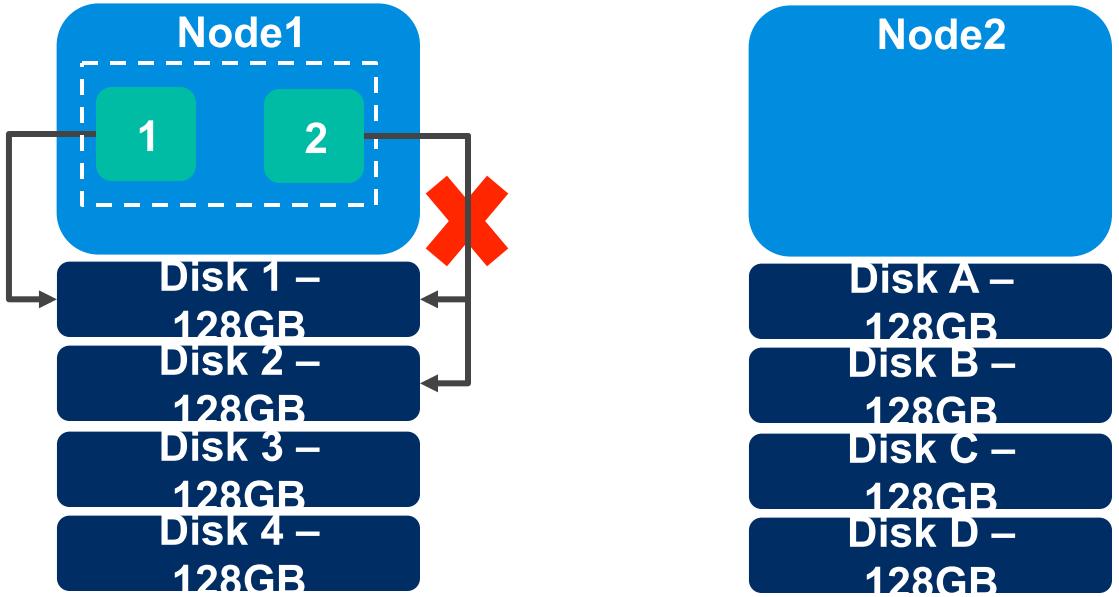
1. Container Rescheduling

- disk has to move
- or rescheduled on same node

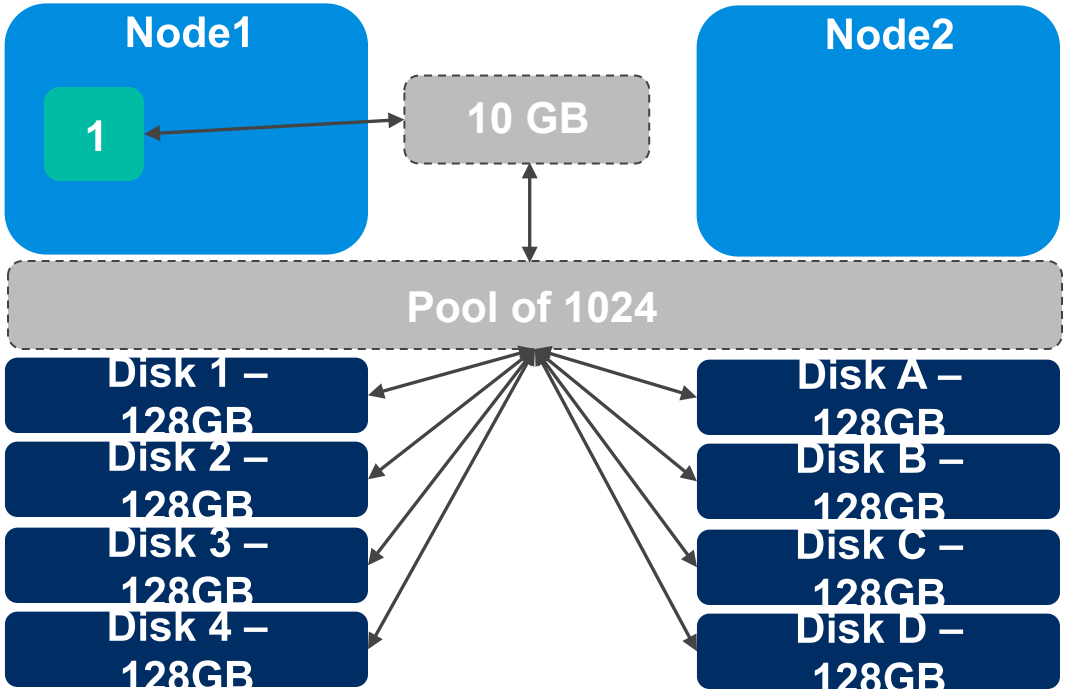


Challenges with Attached / Managed Disks

1. Container Rescheduling
   - disk has to move
   - or rescheduled on same node
2. Container/Disk Challenges
   - schedule all nodes together
   - or 1:1 container:disk relationship
3. Max number of disks/VM



Pooling Disks



1

Portworx

- Pooled software-defined storage solution
- Storage virtualization- serves virtual volumes
- Enterprise grade
  - Backup
  - Snapshots
- Container-focused
  - Docker volume driver
  - Scheduler aware
  - Per-volume encryption
  - If the scheduler moves a container, the volume moves with it

portworx

Demo – Running at Scale

Rob Bagby

We will illustrate enabling:

What

1. Development
2. Running at scale
   - Big Data Solutions
   - Data Persistence
3. Managing at scale
   - Autoscaling
   - Workflows

How

Containers

Orchestrator

DC/OS

Portworx

Workflow Solution

VAMP

CANARY RELEASING AND AUTOSCALING FOR MICROSERVICE SYSTEMS

VAMP.IO

VAMP Artifacts

Static

- Breeds – Describe entities
- Blueprints – Describe topologies
- Scales – Define the size of a deployed service

Runtime

- Deployments – Running Blueprints
- Workflows – NodeJS-bases workflow services
- Gateways – Stable Routing endpoints

Demo – Managing at Scale

Rob Bagby

Session resources

- https://github.com/RobBagby/dcos-kafka-cassandra
- https://github.com/RobBagby/dcos-primer
- http://www.deveducate.com