

How Container Schedulers and Software-Defined Storage will Change the Cloud

David vonThenen
{code} by Dell EMC
@dvonthenen
<http://dvonthenen.com>
github.com/dvonthenen



Agenda

- Review of Software-Defined Storage
- Container Schedulers
- Schedulers + Software-Defined Storage = Awesome!
- To the Cloud!!
- Demo



Review of Software-Defined Storage



What are they?

- Many definitions... most agree on:
- Software-Defined Storage (SDS) serve as abstraction layer above underlying storage
 - Installed software versus Physical Storage Arrays
- Provides a (programmatic) mechanism to provision storage
- Varying degrees of SDS: NFS, VMware VSAN

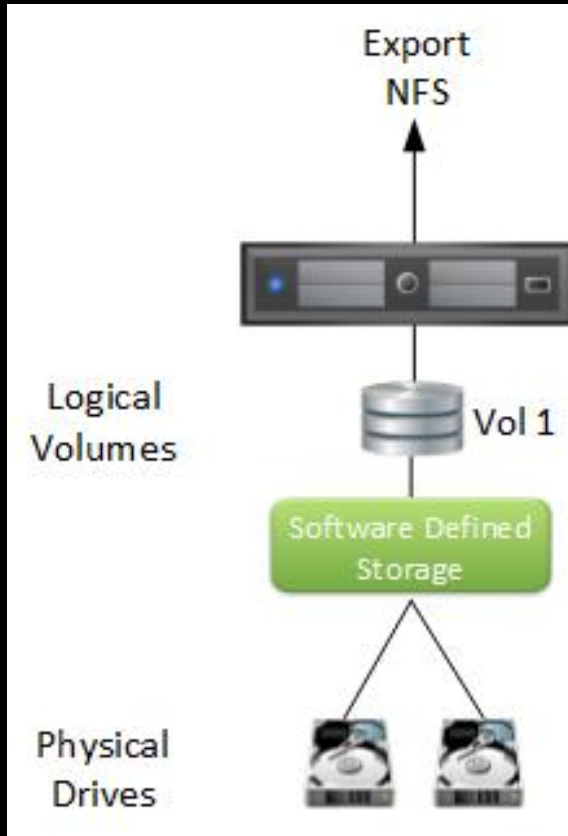


What makes them unique?

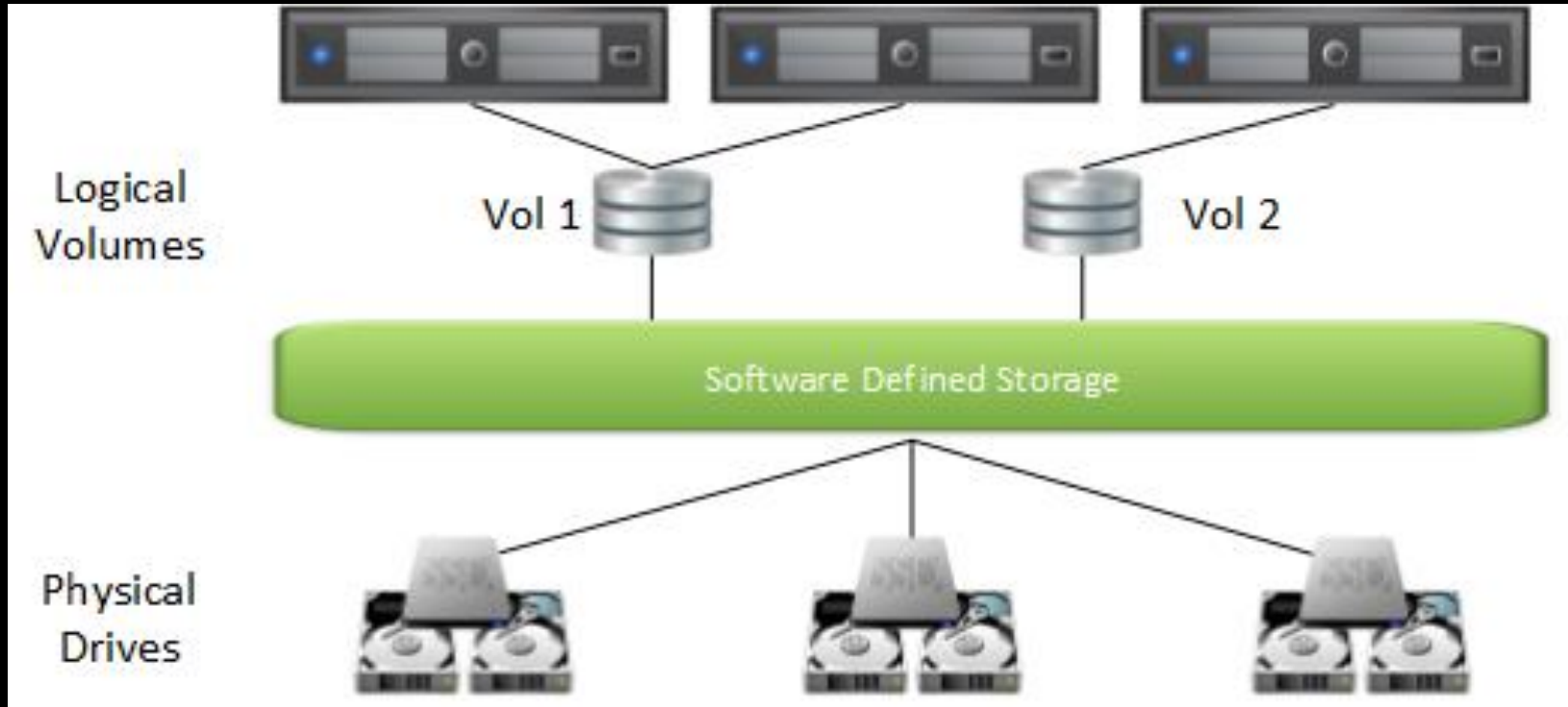
- Operational - Manage provisioning and data independent of underlying hardware
- Physical - Abstract consumed logical storage from underlying physical storage
- Policy - Automation of policy driven both external (users) and internal (platform)
- Day 2 Operations - Maintenance is inherently different



Example: NFS



Example: VSAN

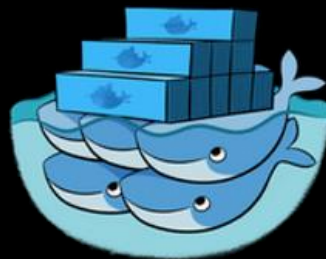


Container Schedulers



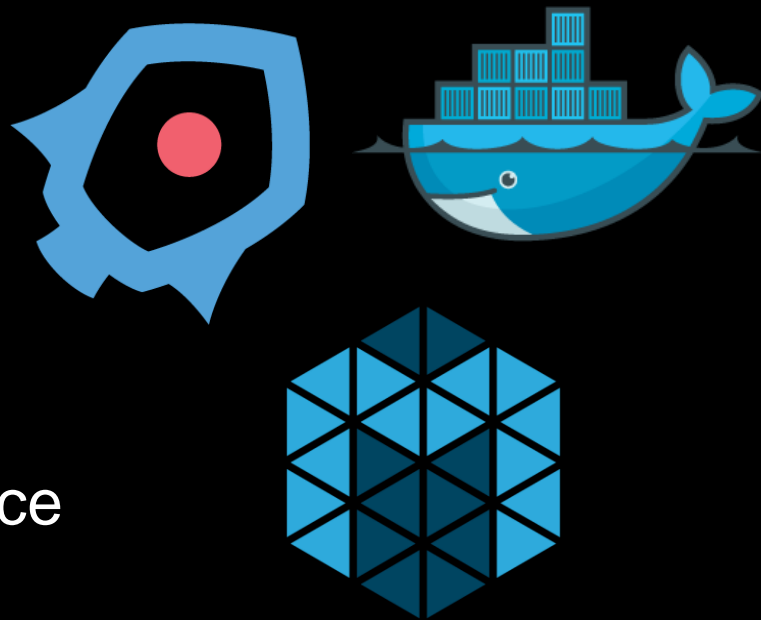
What is a Scheduler?

- Fair and efficient workload placement
- Adhering to a set of constraints
- Quickly (and deterministically) dispatching jobs
- Robust and tolerates errors



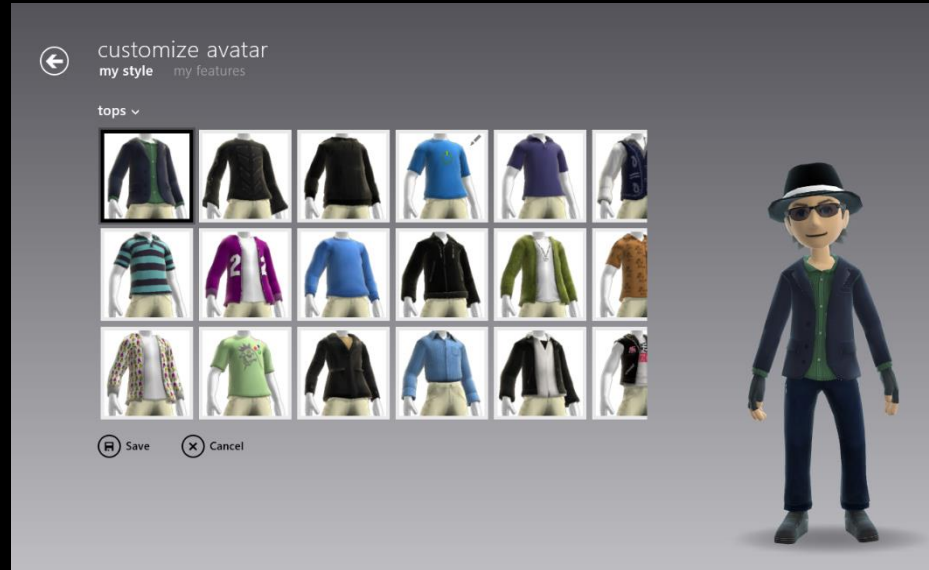
Scheduling Work

- Containers like...
 - Docker
 - Mesos Unified Containerizer
 - rkt (CoreOS)
- Cluster Manager
- Task placement based on resource
- Operational constraints



Custom Scheduling

- Many allow creation of own custom Scheduler
- Customization for your application:
 - Run-Time?
 - Availability?
 - Fault Tolerance?
 - Hardware Accel?
 - Location?



Apache Mesos

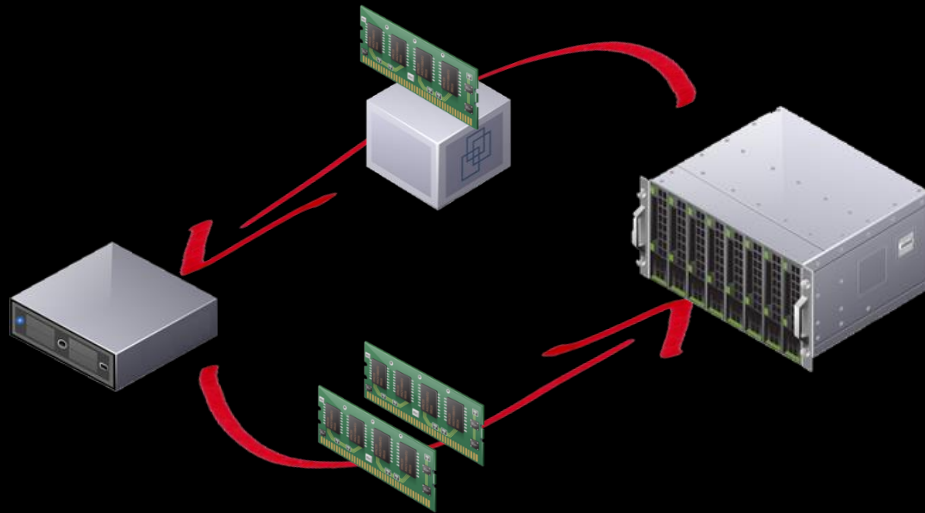


More

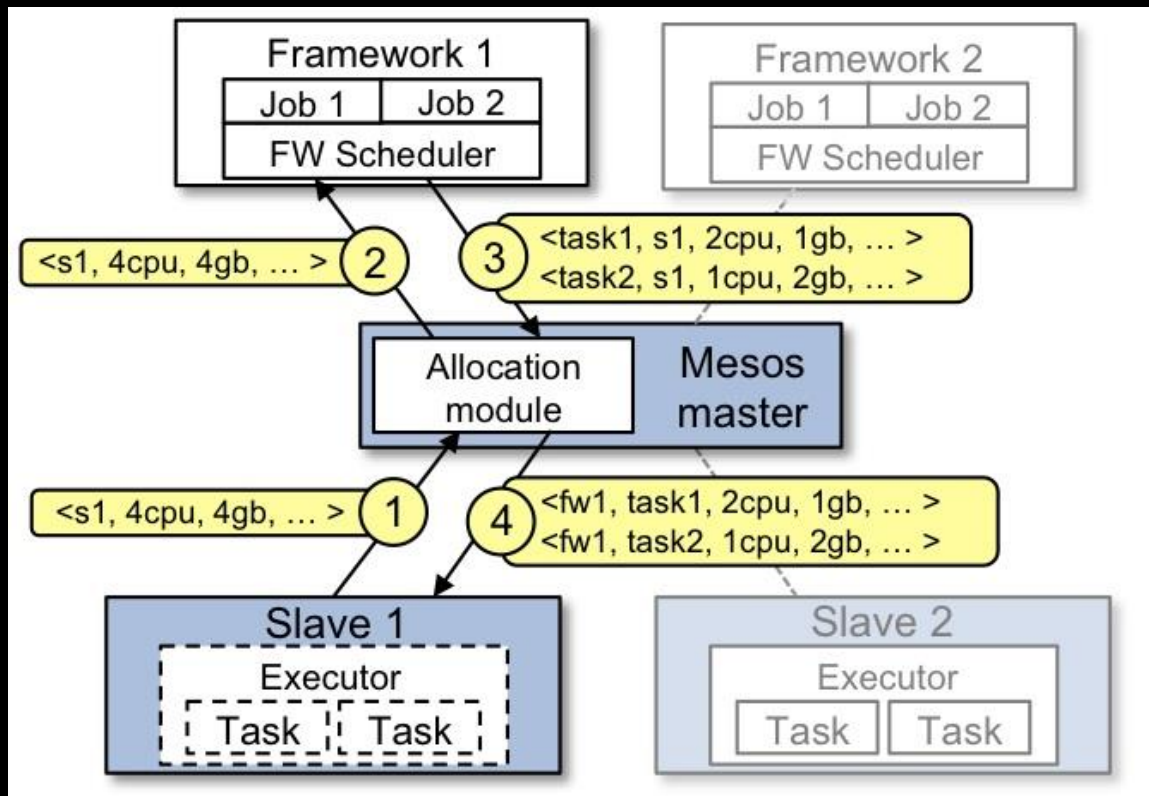


Mesos Frameworks

- Ability to schedule tasks based on Application needs
- Framework implements a Scheduler and Executor
 - Scheduler – Accepts/Denies resources
 - Executor – Application
- Offer / Accept Mechanism
- Multiple Frameworks run within the cluster



Framework / Offer Mechanism



Schedulers and Software Defined Storage



Better Together

- Let's create a Software-Defined Storage Framework
- ScaleIO + Mesos Framework = Awesome Sauce!
- First released in Sept 2016.
Now on version 0.3.1
- <https://github.com/codedellemc/scaleio-framework>



Let's take a look: ScaleIO

- Scale-out block storage
- Linear performance
- Elastic architecture
- Infrastructure agnostic
- Try ScaleIO. It's a free download!

ScaleIO

<https://www.emc.com/products-solutions/trial-software-download/scaleio.htm>















SDS Framework = Mind Blown

- Framework installs and configures Storage Platform on all Scheduler's compute nodes
- Persistent storage **native** to scheduling platform
- Globally accessible storage
- What Storage array? Reduce complexity
- Deploy Anywhere!



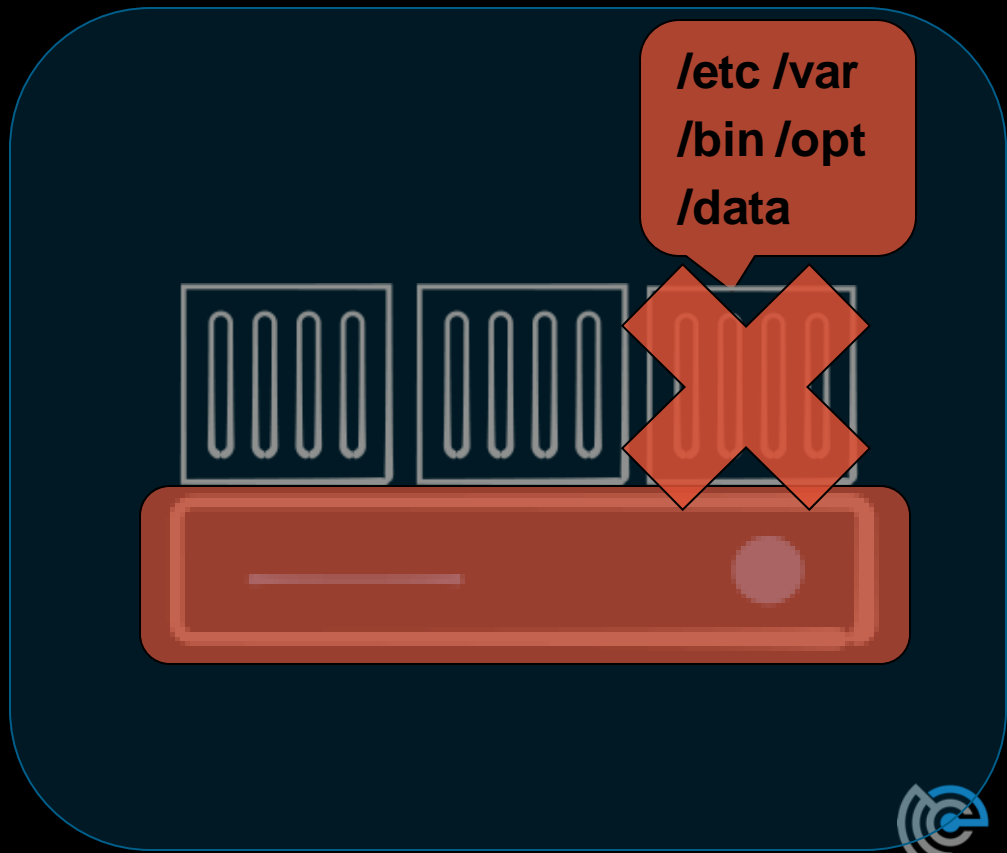
Containers Today

- Many container workloads are long running
- Many have state: user data, configuration, and etc
- Top 7 of 12 Apps in Docker Hub are persistent applications

 nginx official	3.1K STARS	10M+ PULLS	DETAILS
 busybox official	672 STARS	10M+ PULLS	DETAILS
 ubuntu official	4.0K STARS	10M+ PULLS	DETAILS
 registry official	845 STARS	10M+ PULLS	DETAILS
 swarm official	346 STARS	10M+ PULLS	DETAILS
 redis official	2.2K STARS	10M+ PULLS	DETAILS
 mongo official	1.9K STARS	10M+ PULLS	DETAILS
 mysql official	2.4K STARS	10M+ PULLS	DETAILS
 node official	2.2K STARS	10M+ PULLS	DETAILS
 postgres official	2.1K STARS	10M+ PULLS	DETAILS
 elasticsearch official	1.2K STARS	10M+ PULLS	DETAILS
 wordpress official	1.0K STARS	5M+ PULLS	DETAILS

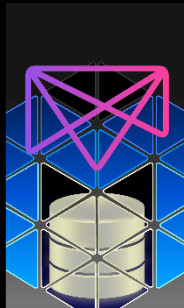
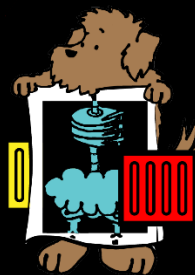
Death of a Container

- Where does my data go?
- Turned to the compute node's local disk to store data
- What happens on a node failure?
- Production applications require high availability
- External Storage!



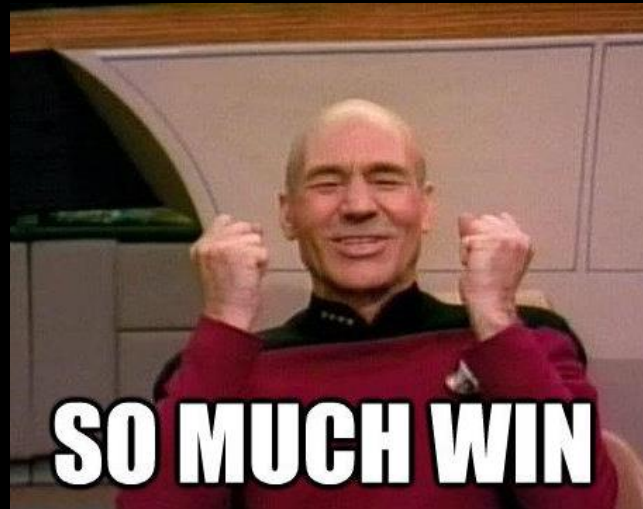
External Storage Enablement

- REX-Ray
 - Vendor agnostic storage orchestration engine
 - AWS, GCE, ScaleIO, VirtualBox, many more
 - <https://github.com/codedellemc/rexray>
- mesos-module-dvdi
 - Hook for Mesos nodes to manage external storage
 - <https://github.com/codedellemc/mesos-module-dvdi>
 - Contributed back to and is apart of Mesos proper



What this Means for your Apps

- Tolerates node failures
- Highly Available containers and Apps!
- Insulates changes with:
 - container scheduler (APIs, etc)
 - storage platform (workflows, APIs, etc)
- Production ready!



To the Cloud!



Moving towards the Cloud

- Applications with management APIs
- Cloud is perfect to enable DevOps
- What makes these cloud accessible?



Self Monitoring Apps

- Framework deploy and configure applications.
- Enable application monitoring via Management APIs
- Determine health and remediate!
- Can fix themselves, but to what end?



Self-aware Applications

- AWS SDK – 10 Language bindings
- Software-based Storage Platform with a Cloud Platform driven by APIs
- Applications that change their environment
 - Maintenance, Remediation, Performance, etc
- Self-aware applications! Skynet!



Premise: Self Managing

- Framework can monitor and self remediate Software-based Storage Platform
- The Scenario:
 - ScaleIO has a Storage Pool that is approaching full
 - Identifies the health check warning
 - Creates new EBS volumes in EC2 to expand the Storage Pool



Demo



Configuration

- Mesos Configuration
 - 3 Node Mesos Cluster (Management)
 - 3 Mesos Agent nodes (Compute)
- ScaleIO Cluster (Scale-out storage)
 - Will install on top of 3 Mesos Agent nodes
 - 180 GB local disks on each node to make up this Storage Pool

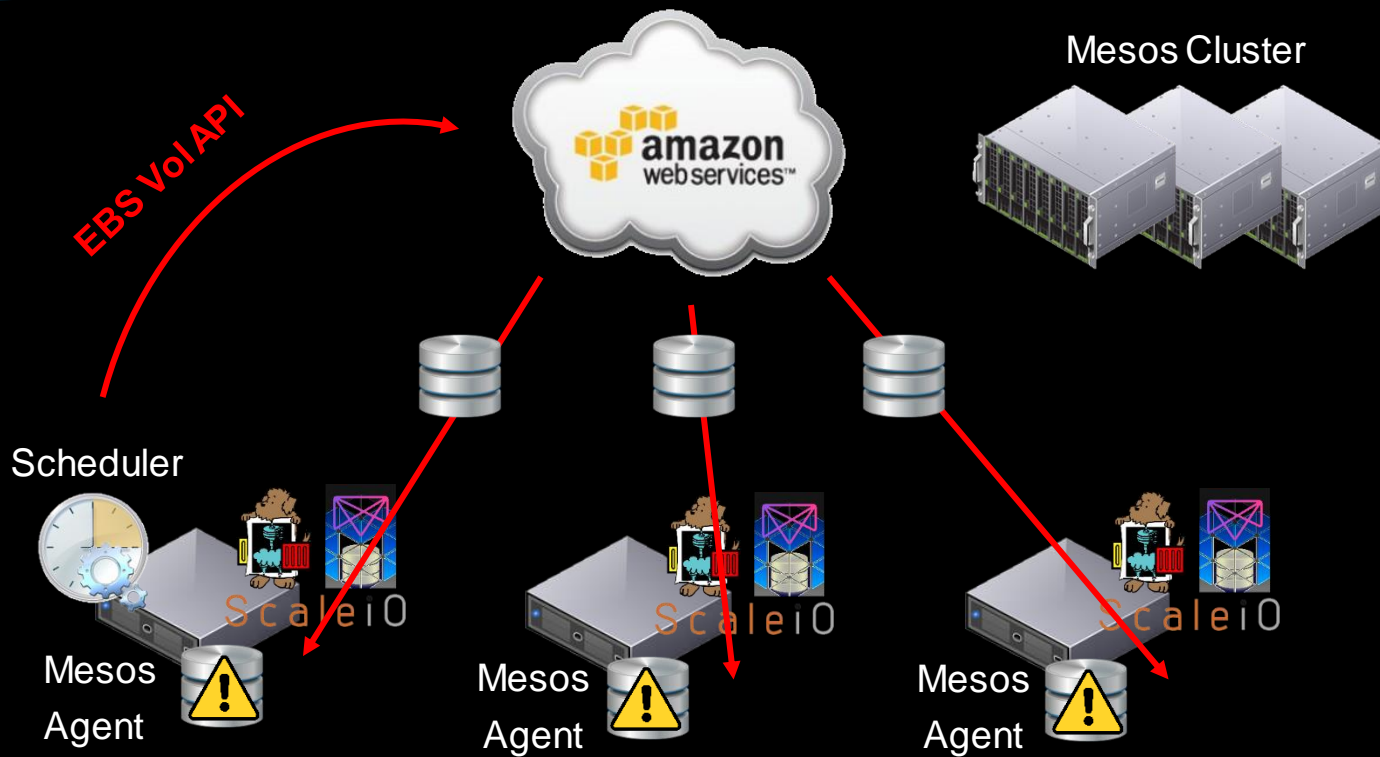


Configuration (Cont.)

- ScaleIO Framework
 - GitHub: <https://github.com/codedellemc/scaleio-framework>
- Persistent External Storage
 - Using REX-Ray
 - › GitHub: <https://github.com/emccode/rexray>
 - Using mesos-module-dvdi
 - › GitHub: <https://github.com/emccode/mesos-module-dvdi>



The Moving Parts



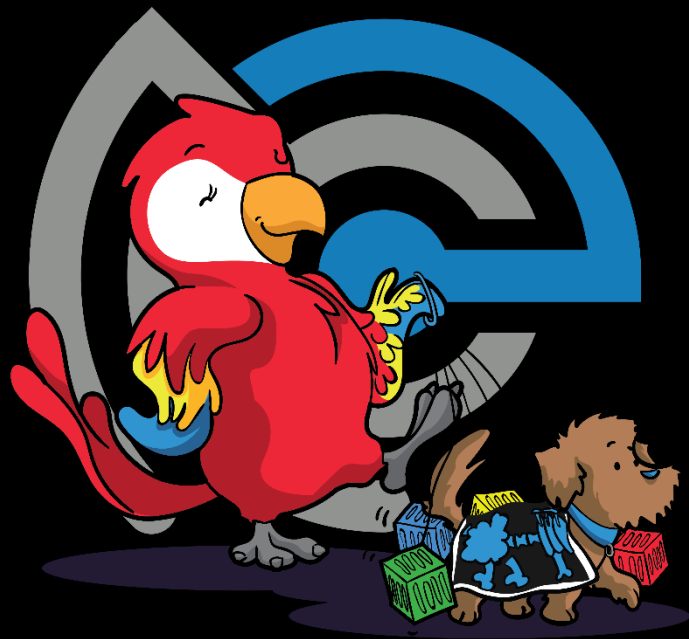
#CodeOpen



Thank you

codedellemc.com

#CodeOpen



codedellemc.com