An Introduction to Apache Mesos

Timothy St. Clair
@timothysc

Special Thanks To:
Paco Nathan
http://liber118.com/pxn
@pacoid

Let's have some fun!
Please ask questions.

Overview

- Project Synopsis
- History
- Ecosystem Questions
- More Detailed Description
- Interesting Use Cases
- Current “Shiny” Developments

What is Apache Mesos?

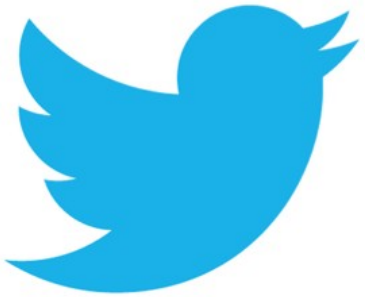
Apache Mesos is a **cluster manager** that provides efficient **resource isolation** and **sharing** across **distributed applications**, or **frameworks**.

It can run Hadoop, MPI, Hypertable, Spark, Elastic Search, Storm, Aurora, Marathon ... and other applications on a dynamically shared pool of nodes.

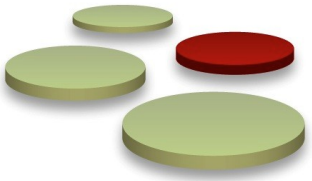
Project Highlights

- Top-level Apache project ~ 1 year (mesos.apache.org)
- Scales to 10,000s of nodes
- Obviates the need for virtual machines
- Isolation for CPU, RAM, I/O, FS, etc.
- Fault-tolerant leader election based on Zookeeper
- API's in C++, Java/Scala, Python, Go, Erlang, Haskell.
- Web UI for inspecting state
- Available for Linux, OpenSolaris, Mac OSX

Who is using Mesos? - Early Adopters





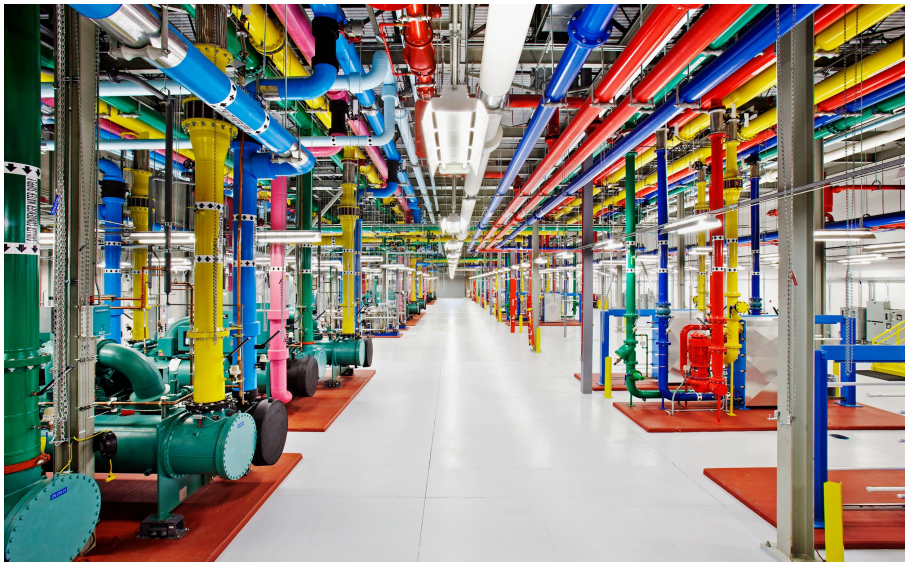


History

Understanding of Datacenter Computing

Google has been doing data center computing for years to address the complexities of large-scale data workflows:

- Leveraging the modern kernel isolation. (cgroups)
- Containerization !Virtualization (lmctfy - Docker)
- Most (>80) jobs are batch jobs, but the majority of resources(55-80%) are allocated to **service jobs.**
- Mixed workloads, multi-tenancy
- Relatively high utilization rates
- JVM? Not so much...
- Reality: scheduling batch is simple;
  - scheduling services is hard/expensive.



Refs.

- The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines
  - http://research.google.com/pubs/pub35290.html
- GAFS Omega John Wilkes
  - https://www.youtube.com/watch?v=0ZFMlO98Jkc
- Taming Latency Variability and Scaling Deep Learning
  - http://youtu.be/nK6daeTZGA8

Ecosystem Questions?

Aren't There Several Other Existing Solutions? - “Sounds Like” a grid

- IBM Platform Symphony
- Microsoft Autopilot
- Univa Grid Engine (SGE)
- Condor (Full Disclosure)

Follow Up:
What is the Gap?

- Many existing grid-solutions had architectural deficiencies around a constraining model. Everything is a “job”.
  - Good at Batch, but tough for Services
  - What happens when you want to write your own distributed application? (no primitives)
  - What happens when you want to write your own scheduler (elastic service). Square wheel reinvention.

What about Clouds?

- Can't you just “Cloud All the things”
  - It's not very efficient
  - Can be quite costly
- It doesn't actually solve the root cause of many of the problems in applications, and in some cases a Cloud can cause more issues, not less.
  - Network Latency
  - Data Gravity
  - Multi-tenant Service Elasticity

    ...

Reality Check

The New Reality

- New applications need to be:
  - Fault Tolerant (Withstand failure)
  - Scalable (Doesn't crumble under it's own weight)
  - Elastic (Can grow and shrink based on demand)
  - Multi-tenent (It can't have it's own dedicated cluster)
- So what does that really mean?

Distributed Applications

- “There's Just No Getting Around It: You're Building a Distributed System” Mark Cavage
  - **queue.acm.org/detail.cfm?id=2482856**
- Key takeaways on architecture:
  - Decompose the business applications into discrete services on the boundaries of fault domains, scaling, and data workload.
  - Make as many things as possible stateless
  - When dealing with state, deeply understand CAP, latency, throughput, and durability requirements.

“Without practical experience working on successful—and failed—systems, most engineers take a "hopefully it works" approach and attempt to string together off-the-shelf software, whether open source or commercial, and often are unsuccessful at building a resilient, performant system. In reality, building a distributed system requires a methodical approach to requirements along the boundaries of failure domains, latency, throughput, durability, consistency, and desired SLAs for the business application at all aspects of the application.”

Emerging at Berkeley

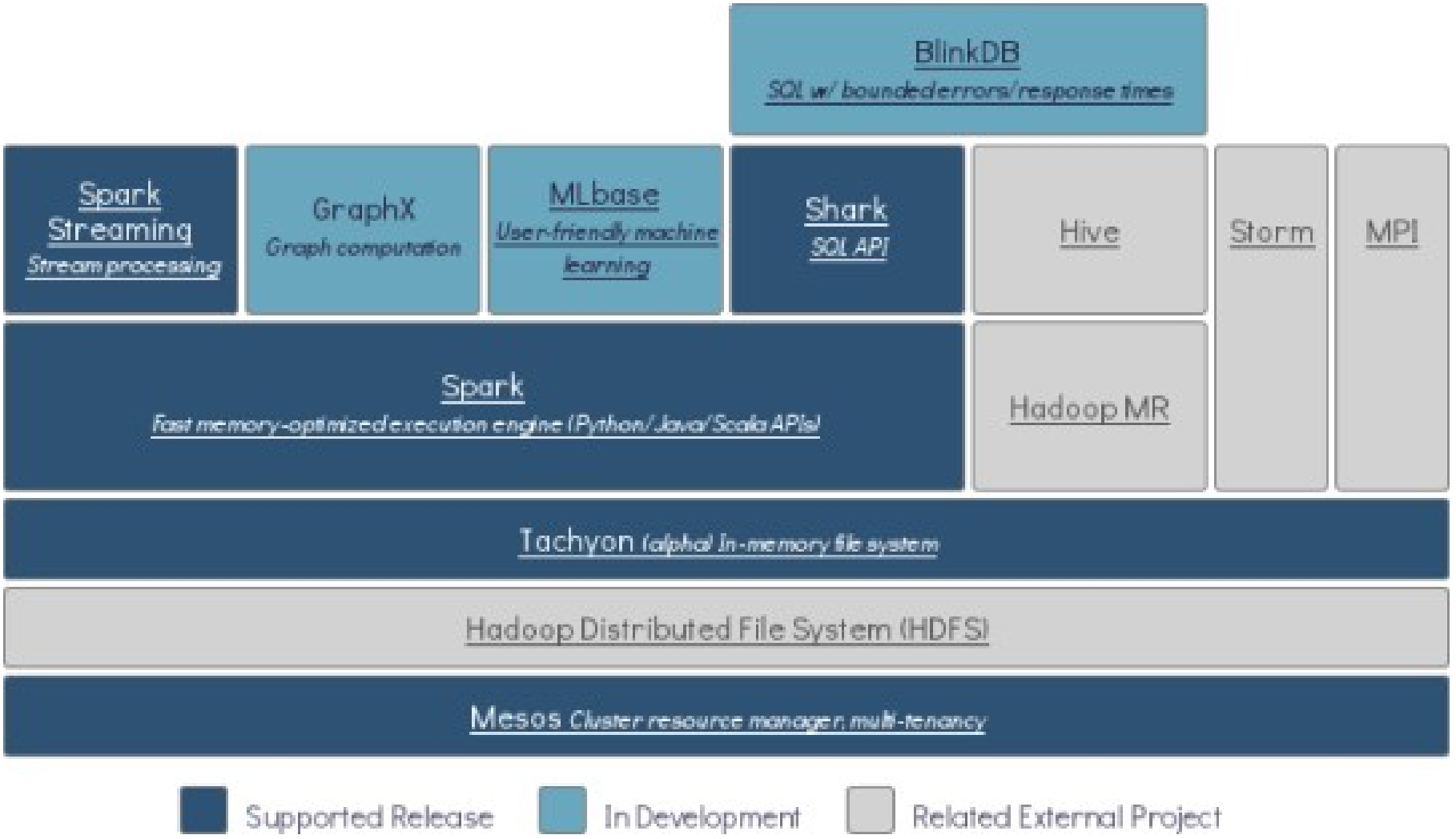
Beyond Hadoop

Hadoop – an open source solution for fault-tolerant parallel processing of batch jobs at scale, based on commodity hardware... However, other priorities have emerged for analytics lifecycle:

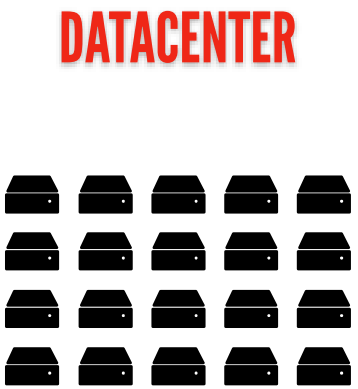
- Applications require integration beyond Hadoop
- Multiple topologies, mixed workloads, mult-tenancy
- Higher utilization
- Lower latency
- High availability
- More then “Just JVM” - e.g. Python ...

Next Generation Data Analytics Stack

BDAS Stack

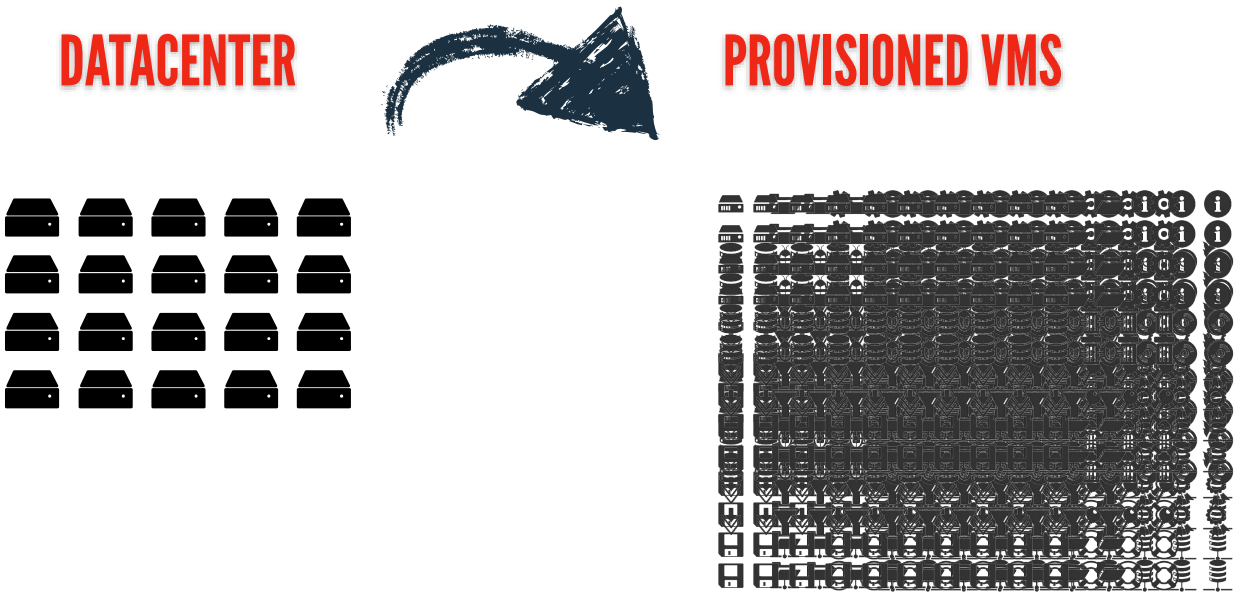

BlinkDB
SQL w/ bounded errors/response times
Spark Streaming
Stream processing
GraphX
Graph computation
MLbase
User-friendly machine learning
Shark
SQL API
Hive
Storm
MPI
Spark
Fast memory-optimized execution engine (Python/Java/Scala APIs)
Hadoop MR
Tachyon (alpha) In-memory file system
Hadoop Distributed File System (HDFS)
Mesos Cluster resource manager, multi-tenancy
Supported Release
In Development
Related External Project

Prior Practice: Dedicated Servers



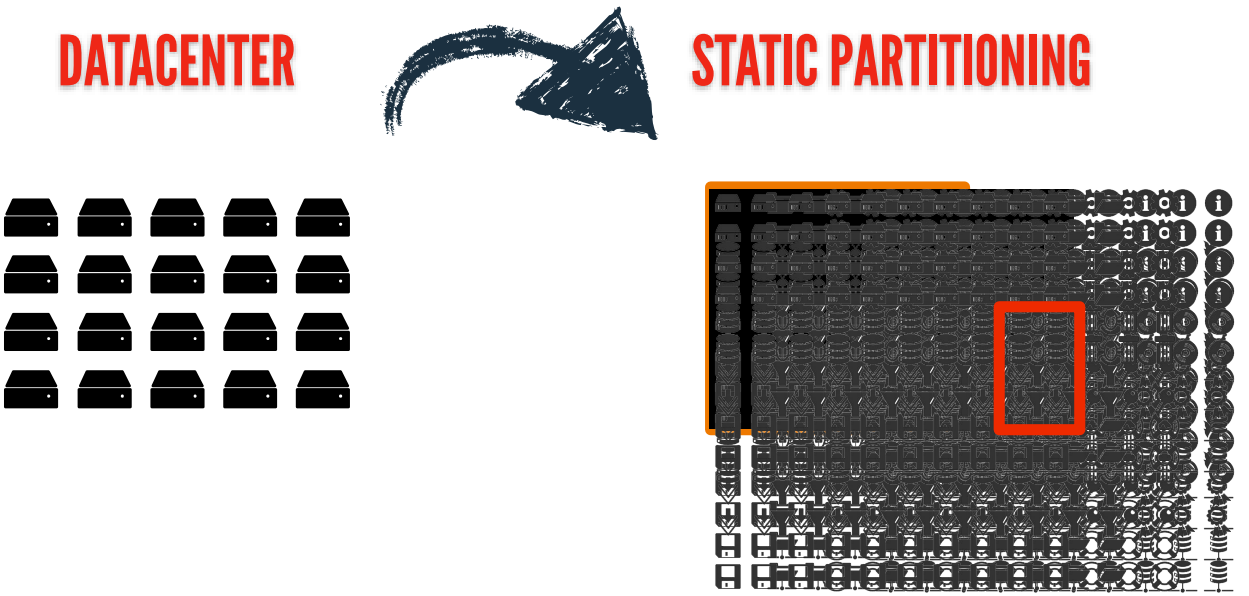
- *low utilization rates*
- *longer time to ramp up new services*

Prior Practice: Virtualization



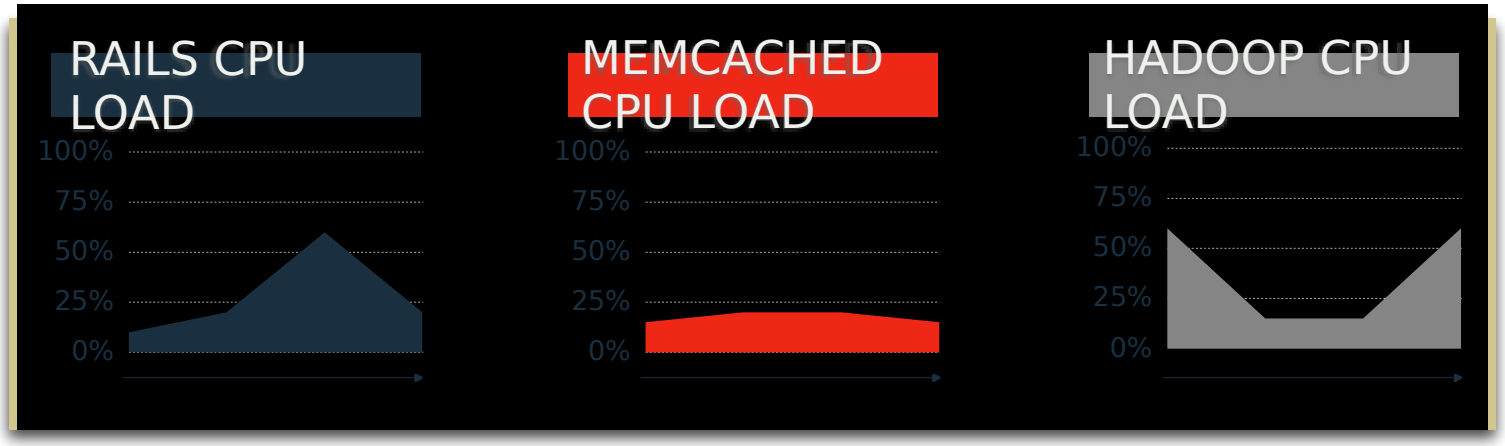
- *even more machines to manage*
- *substantial performance decrease due to virtualization*
- *VM licensing costs*

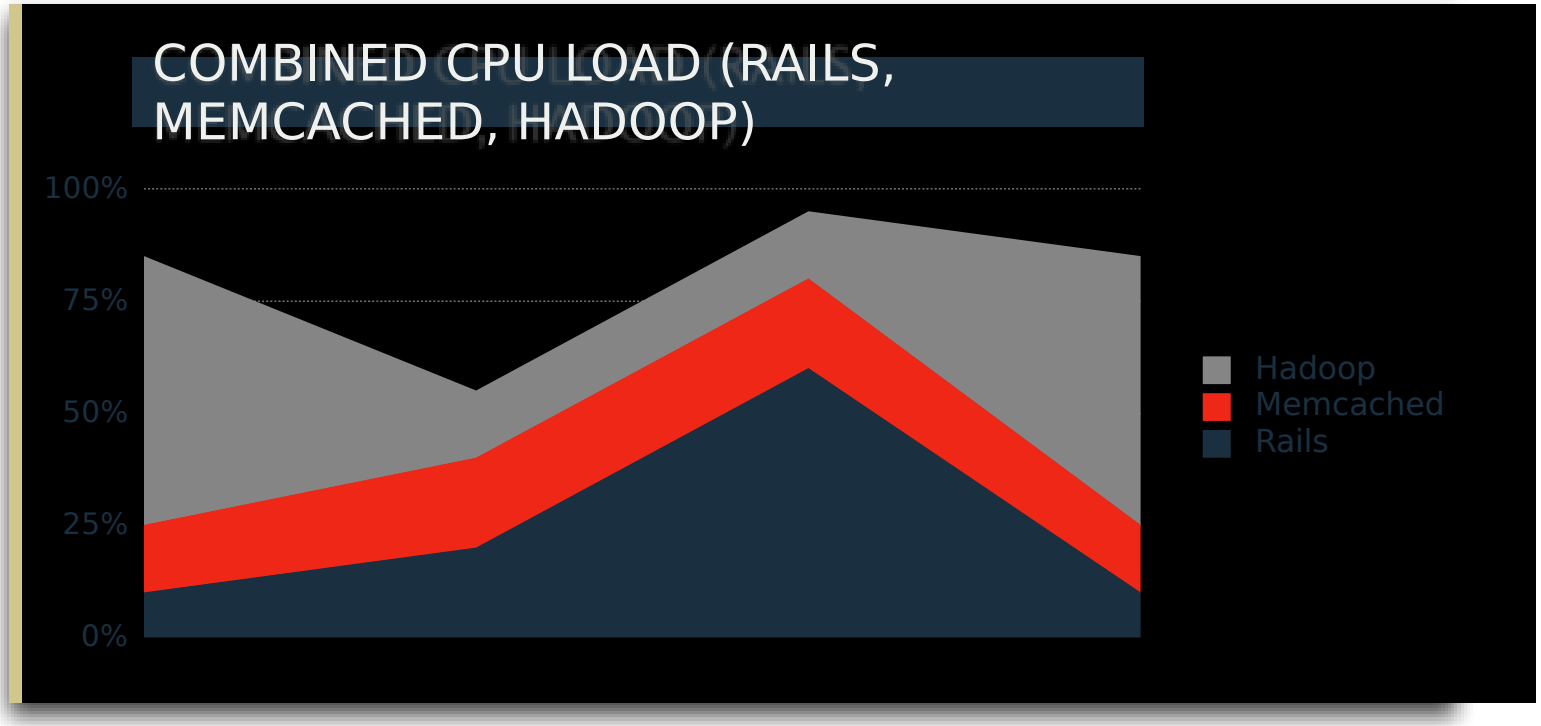
Prior Practice: Static Partitioning



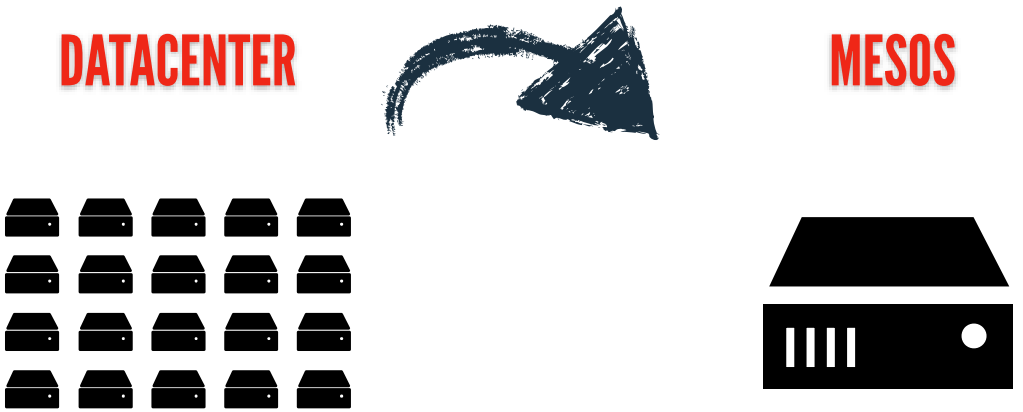
- *even more machines to manage*
- *substantial performance decrease due to virtualization*
- *VM licensing costs*
- *failures make static partitioning more complex to manage*

What are the costs of Single Tenancy?





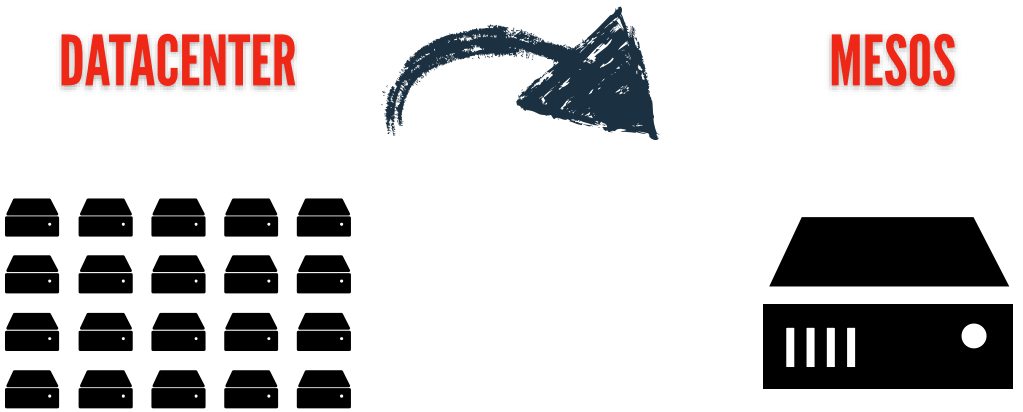
Mesos: One Large Pool of Resources



“We wanted people to be able to program for the datacenter just like they program for their laptop."

Ben Hindman

Mesos: One Large Pool of Resources

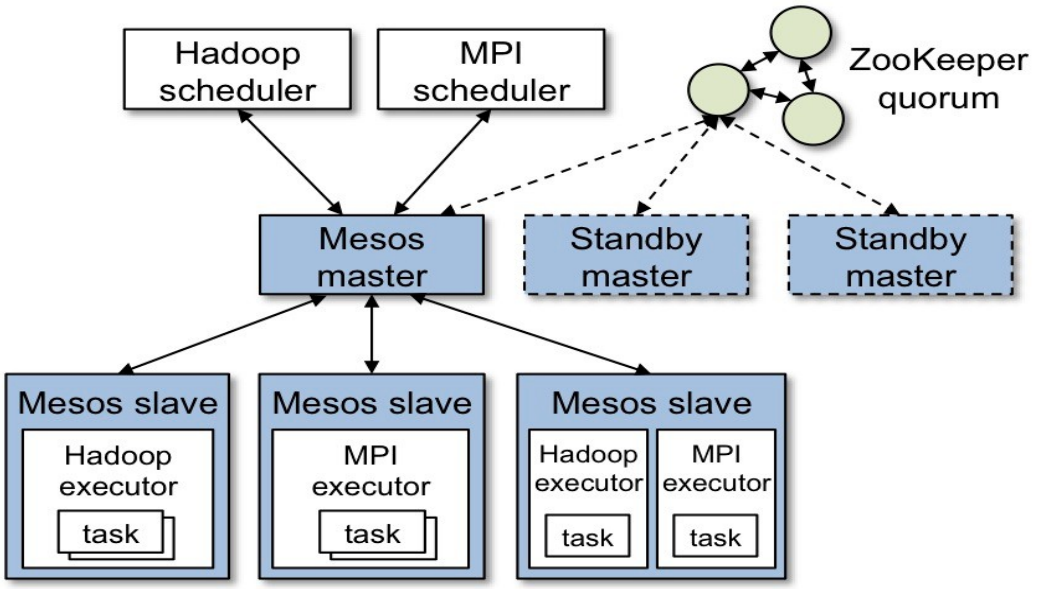


“We wanted people to be able to program for the datacenter just like they program for their laptop."

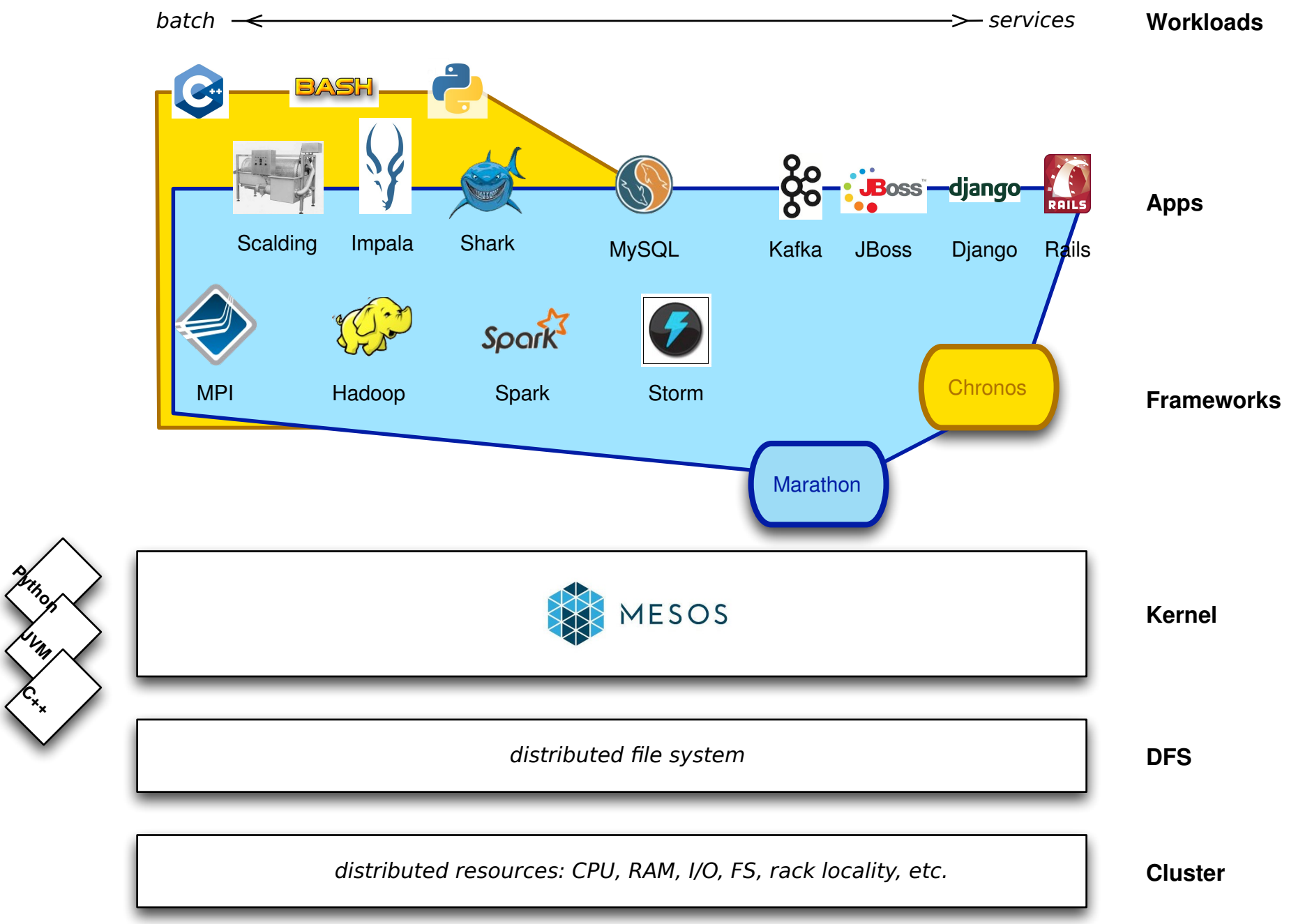
Ben Hindman

Re-eval – What is Mesos?

- Mesos is a meta-scheduler
  - Mesos is a distributed system to build and run distributed systems.
- Microkernel for the datacenter.
  - Common substrate, or programming abstractions, for creating, or adapting distributed applications.



Mesos – architecture



Use Cases

Case Study: Twitter (bare metal / on premise)

“Mesos is the cornerstone of our elastic compute infrastructure – it’s how we build all our new services and is critical for Twitter’s continued success at scale. It's one of the primary keys to our data center efficiency."

Chris Fry, SVP Engineering

blog.twitter.com/2013/mesos-graduates-from-apache-incubation

wired.com/gadgetlab/2013/11/qa-with-chris-fry/

- key services run in production: analytics, typeahead, ads
- Twitter engineers rely on Mesos to build all new services
- instead of thinking about static machines, engineers think about resources like CPU, memory and disk
- allows services to scale and leverage a shared pool of servers across datacenters efficiently
- reduces the time between prototyping and launching

Case Study: Airbnb (fungible cloud infrastructure)

“We think we might be pushing data science in the field of travel more so than anyone has ever done before… a smaller number of engineers can have higher impact through automation on Mesos."

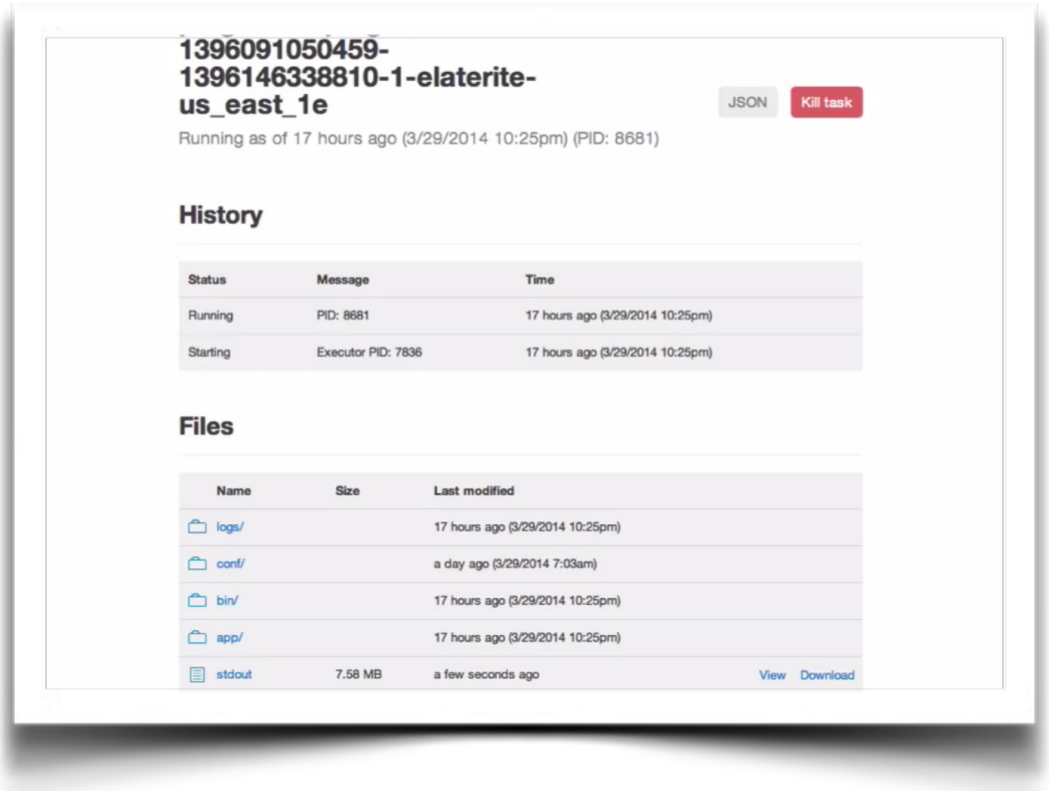
Mike Curtis, VP Engineering
gigaom.com/2013/07/29/airbnb-is-engineering-itself-into-a-data...

- improves resource management and efficiency
- helps advance engineering strategy of building small teams that can move fast
- key to letting engineers make the most of AWS-based infrastructure beyond just Hadoop
- allowed company to migrate off Elastic MapReduce
- enables use of Hadoop along with Chronos, Spark, Storm, etc.

Case Study: HubSpot (cluster management)

Tom Petr
youtu.be/ROn14csiikw

- 500 deployable objects; 100 deploys/day to production; 90 engineers; 3 devops on Mesos cluster
- “Our QA cluster is now a fixed $10K/month — that used to fluctuate”



The Shiny!

Dock-ah, dock-ah, dock-ah

- Enumerate hyperbolic awesome-sauce!
- Mesos had plugable add-ons for docker, now they are being 1st classed into the core.
- Google Kubernetes Framework: http://gigaom.com/2014/06/11/why-google-is-sowing-the-seeds-of-container-based-computing/

API's & More Plugability

- Always supported Protobuf API's compiled against core libraries. Now native bindings are occurring in the wild. (Go, Python, Java).
- Additional Plug-ins for Containerizers, Isolators, ...

Want to know More?
Come Join Us @ #MesosCon

Thank You!

mesos.apache.org

@timothysc