cloudera

ENTERPRISE MACHINE LEARNING ON K8S: LESSONS LEARNED AND THE ROAD AHEAD

Tim Chen, Cloudera
Tristan Zajonc, Cloudera

WHO ARE WE?

Tim Chen
Cloudera
@tnachen

Lead for Cloudera Machine Learning
Former Hyperpilot CEO/Cofounder
Apache PMC for Drill, Mesos

Tristan Zajonc
Cloudera
@tristanzajonc

CTO for Machine Learning at Cloudera
Former Sense CEO/Cofounder

DISCLAIMER

The information in this document is proprietary to Cloudera. No part of this document may be reproduced or disclosed without the express prior written permission of Cloudera.

This document is not binding upon Cloudera in any way (including with respect to Cloudera's course of business, product strategy, future releases, or development commitments), and is not a part of or subject to any license agreement or other agreement you may have with Cloudera. Cloudera makes no commitments about any future developments or functionality in any Cloudera product. The development, release, and timing of release of any software features or functionality described in this document remains at the discretion of Cloudera and you should not rely on any statements about development plans or anticipated functionality in this document when making any purchasing decisions.

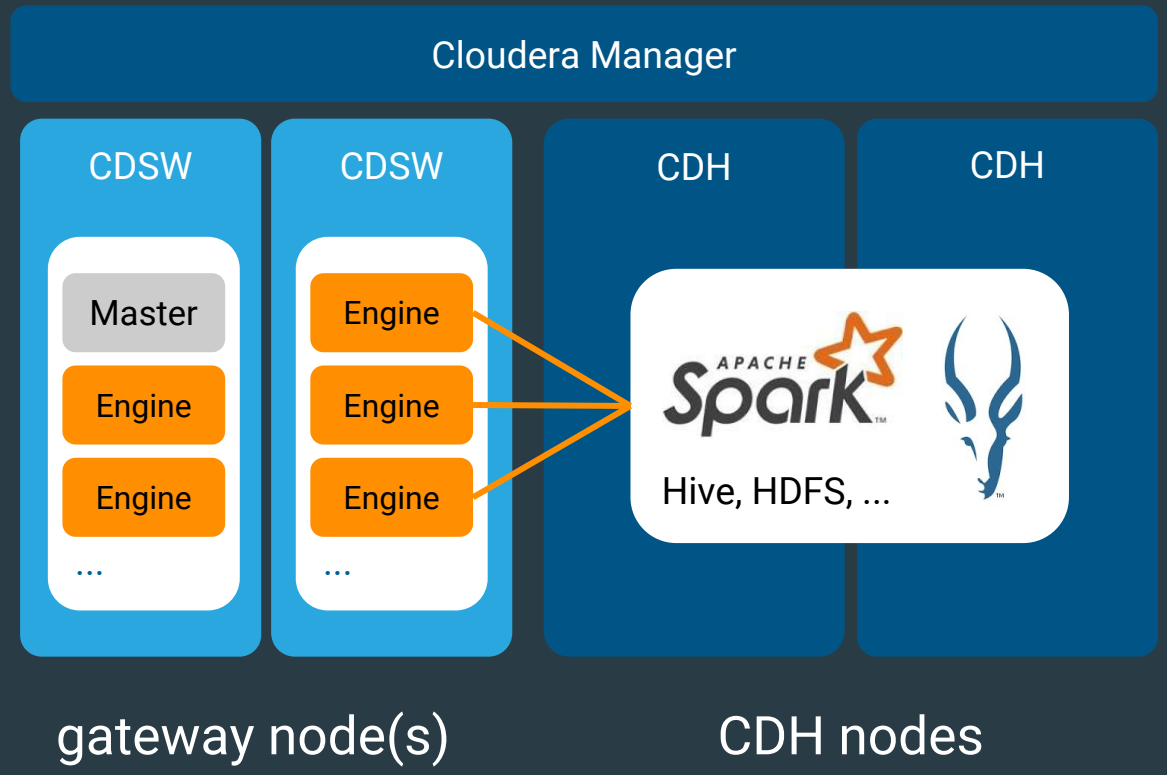
Cloudera assumes no responsibility for errors or omissions in this document. Cloudera does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose or non-infringement. Cloudera shall have no liability for damages of any kind including without limitation direct, special, indirect or consequential damages that may result from the use of these materials."

cloudera + 

CLOUDERA DATA SCIENCE WORKBENCH

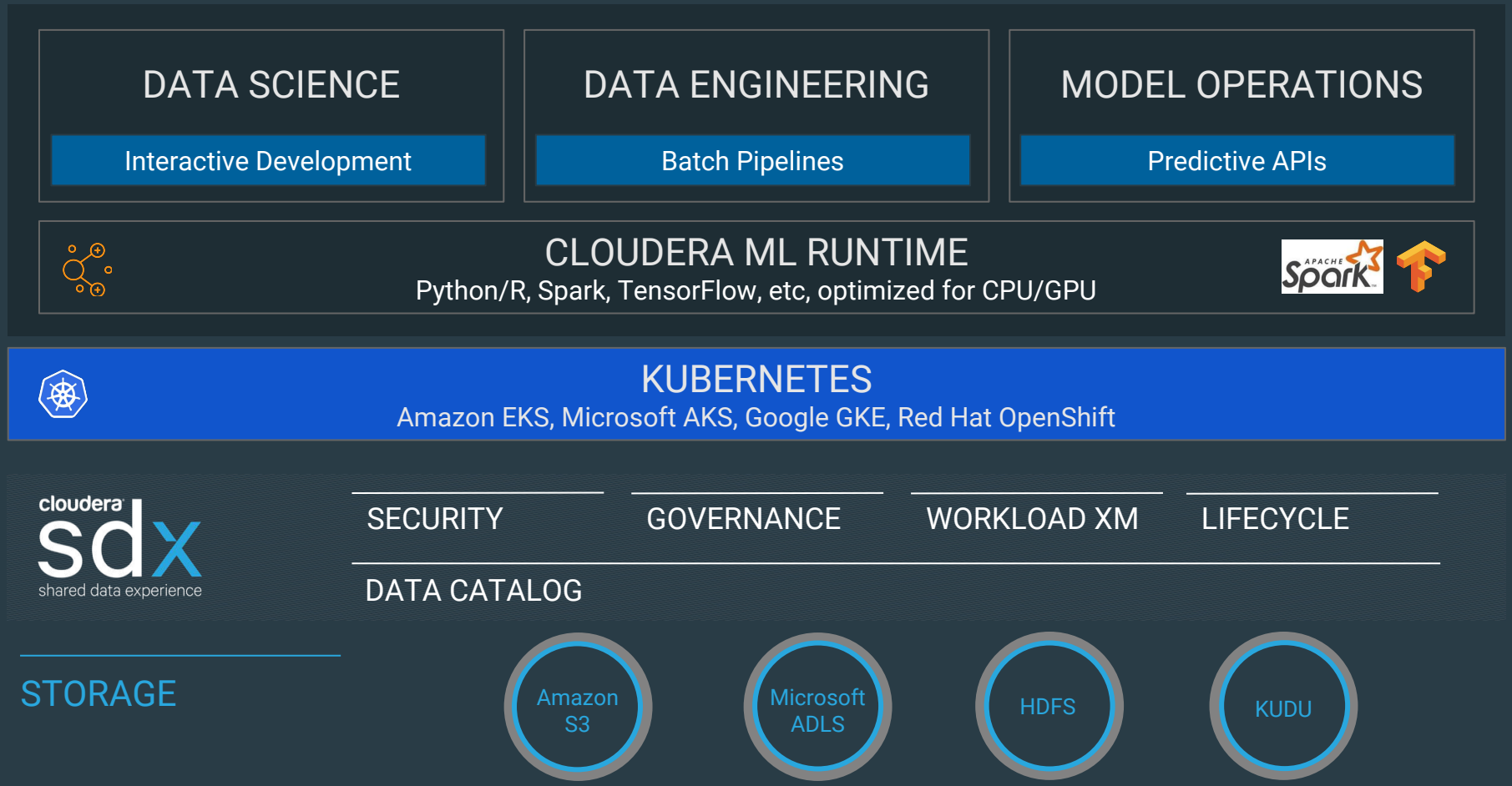
Collaborative data science experience powered by Kubernetes

- Built with Docker and Kubernetes
  - Isolated, reproducible user environments
- Supports both big and small data
  - Local Python, R, Scala runtimes
  - Connect to any data source
  - Schedule & share GPU resources
  - Scale to CDH with Spark, Impala, Hive
- Secure and governed by default
  - Easy, audited access to Kerberized clusters
  - Leverages shared platform services
- Deployed with Cloudera Manager



CLOUDERA MACHINE LEARNING (PREVIEW)

Cloud-native machine learning platform for the enterprise



End-to-end ML lifecycle for teams to build at scale

Rapid provisioning and elastic autoscaling with multi-cloud portability powered by Kubernetes

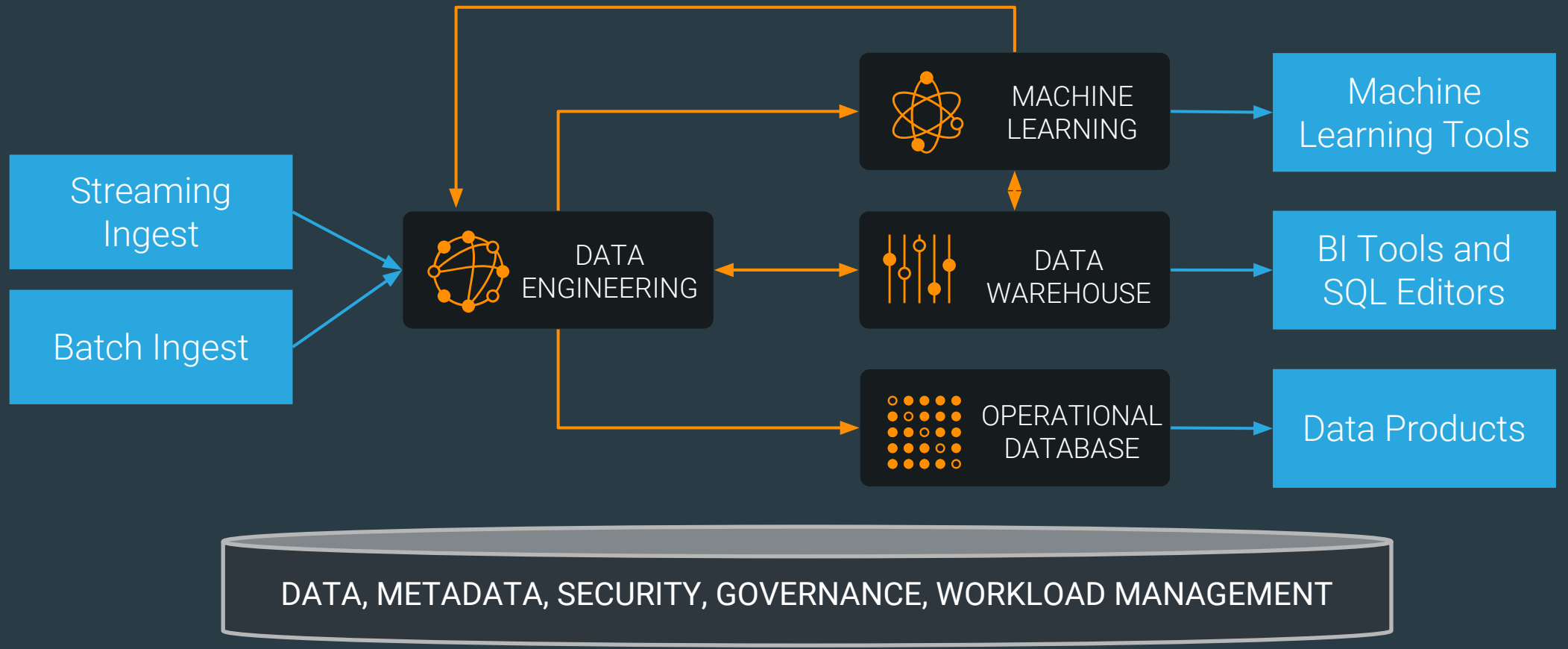
Containerized workloads for open data science with easy dependency management

Distributed GPU training for accelerated deep learning

Secure data access to HDFS or cloud object storage with shared schema, governance

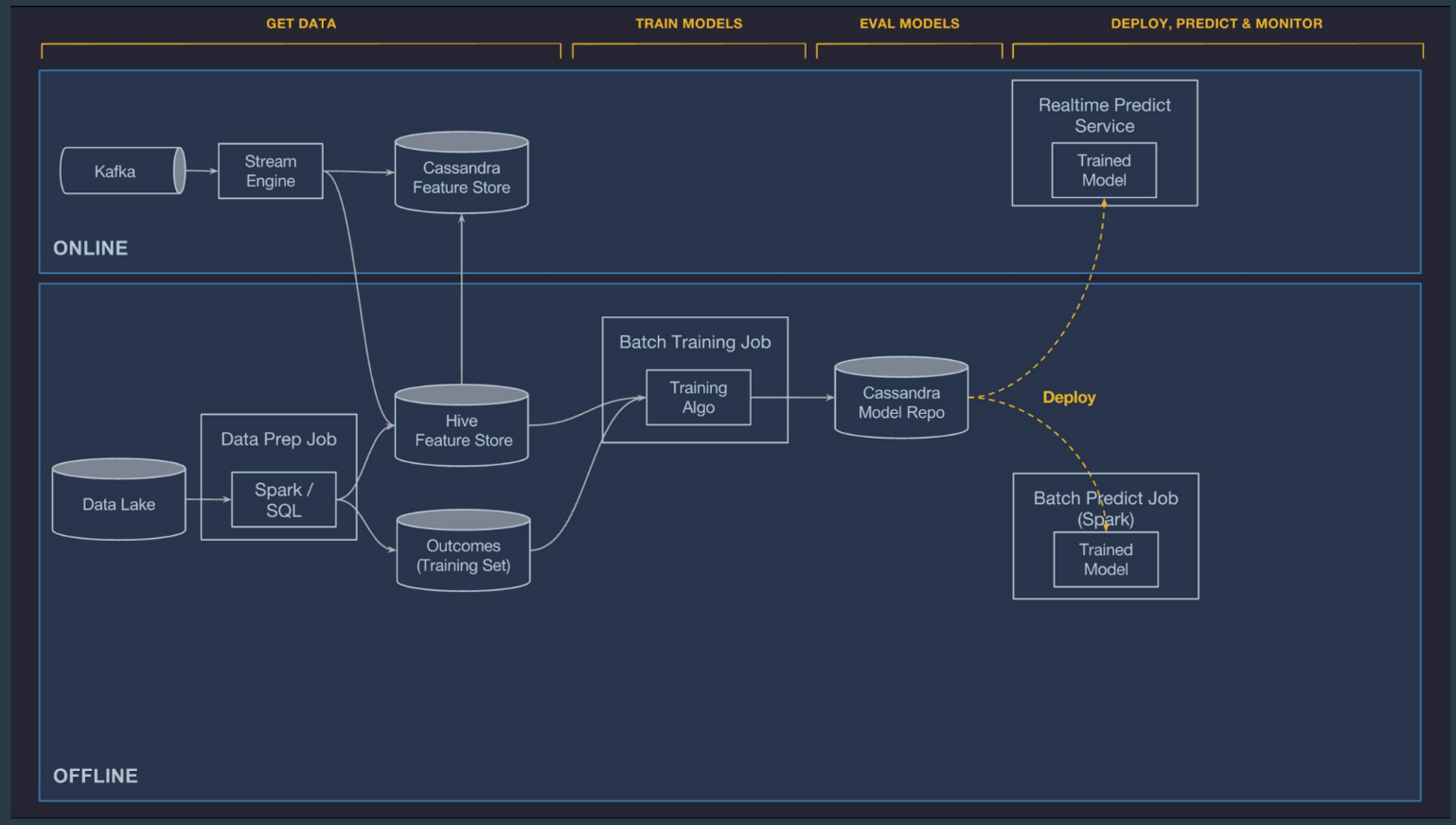
SOME LESSONS LEARNED

LESSON 1: ENTERPRISE ML REQUIRES BIGGER PICTURE



UBER

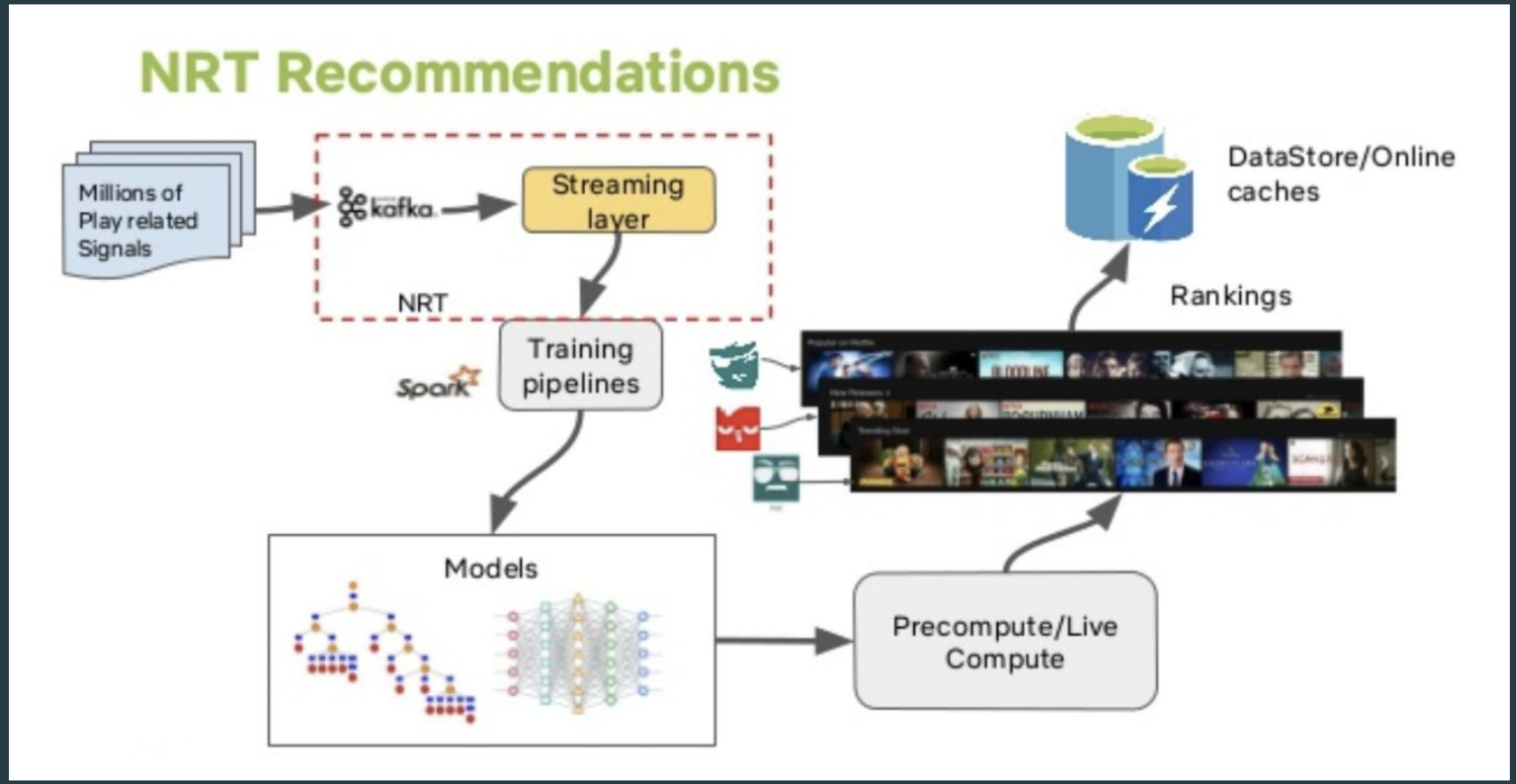
Michelangelo: Uber's Machine Learning Platform



Source: https://eng.uber.com/michelangelo/

NETFLIX

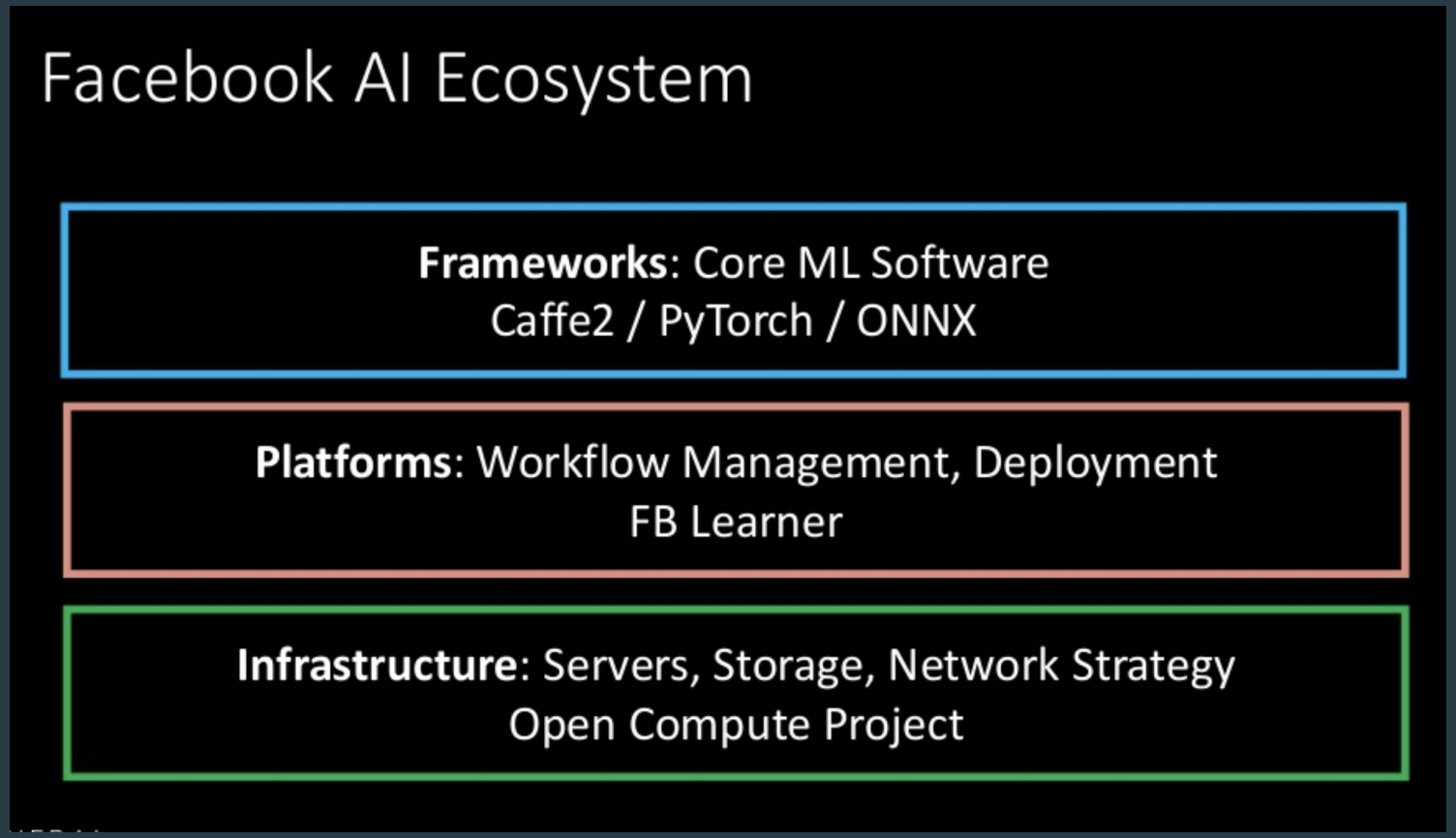
Netflix recommendation infrastructure



Source: https://medium.com/netflix-techblog/netflix-at-spark-ai-summit-2018-5304749ed7fa

FACEBOOK

Facebook's AI infrastructure



Source: https://www.slideshare.net/KarthikMurugesan2/facebook-machine-learning-infrastructure-2018-slides

EMERGING CONSENSUS FOR ENTERPRISE ML AT SCALE

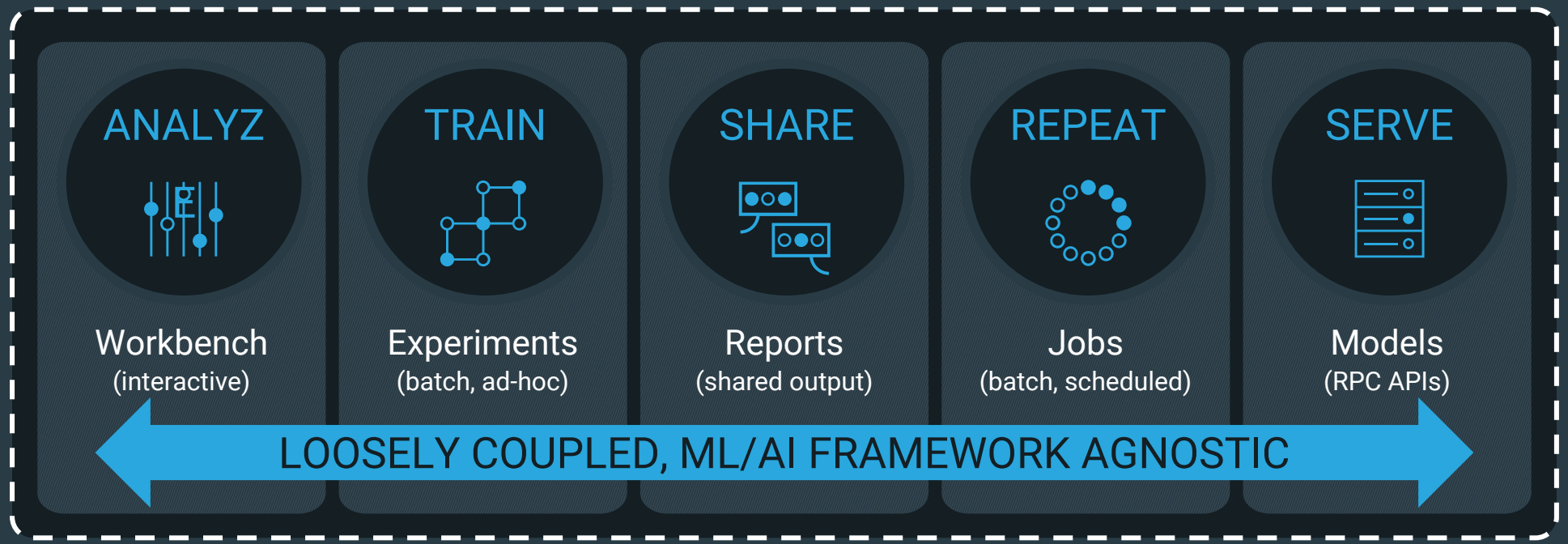
Cloud Native Infrastructure

Big Data Platform

ML/AI Frameworks

LESSON 2: FOCUS ON WORKFLOWS NOT FRAMEWORKS

Data scientists want to use every library under the sun, platform should support that



WHAT DOES THIS LOOK LIKE IN K8S?

Let's talk operators and CRDs….

Cloudera ML is built around one Operator and four "CRDs":

- Session (Interactive)
- Experiment (Batch)
- Job (Scheduled)
- Model (Online API)

No long-lived operators for particular libraries: TFJob, SparkJob, etc

Kubernetes is not exposed to data scientist. Goal is serverless experience.

cldr ml run --cmd “python train.py”

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()
results = sc.parallelize(xrange(0, 1000)) \
   .map(f).collect()
```

ROAD AHEAD

WHY SPARK ON KUBERNETES?

- Unified workflow for distributed data engineering and machine learning
- Simple dependency management, particularly for ML workloads
  - python: pip install, R: install.packages, etc
- Elastic compute (CPU / GPU) in cloud
- Improved utilization and multi-tenancy

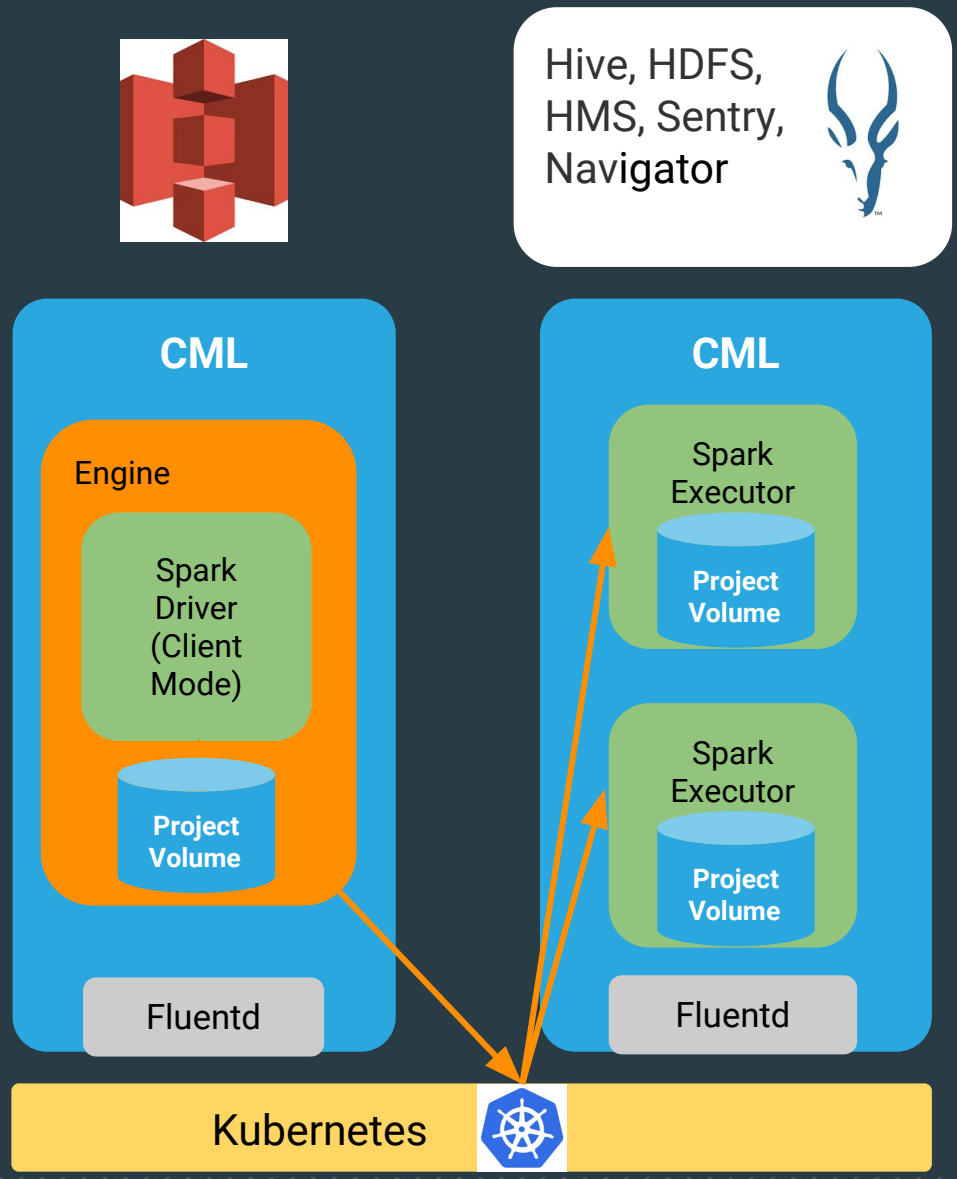
SPARK ON KUBERNETES

- Spark has native integration with Kubernetes since 2.3+
  - Runs Spark fully containerized on K8s
  - Allows Spark to add / remove executors with user specified Spark configuration

- Able to leverages Kubernetes features to facilitate launching Spark (e.g: Node Affinities, RBAC, Namespaces, etc).

SPARK ON KUBERNETES IN CML

- Create service account and namespace for quota
- Configure Spark driver / executor configuration to connect to k8s. Also populate kerberos related information for accessing HDFS and other CDH services.
- Populate dependency management related configurations using pod template: volume, image, user information, etc.
- Fluentd configured to pick up logging and metric information to external storage



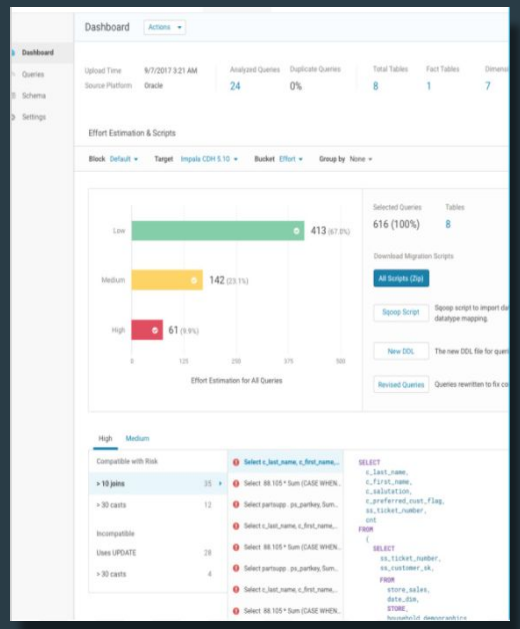
DEMO

SPARK AND K8S UPSTREAM ONGOING WORK

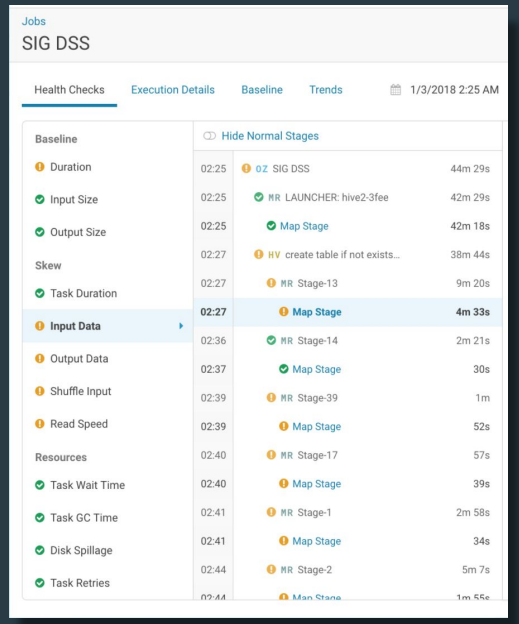
- Kerberos support
- Dynamic allocation
- GPU support
- Scale and automation tests
- Advanced scheduling (FairScheduler, Hierarchical Queues, etc)

AIOPS - AI-DRIVEN OPTIMIZATION AT SCALE

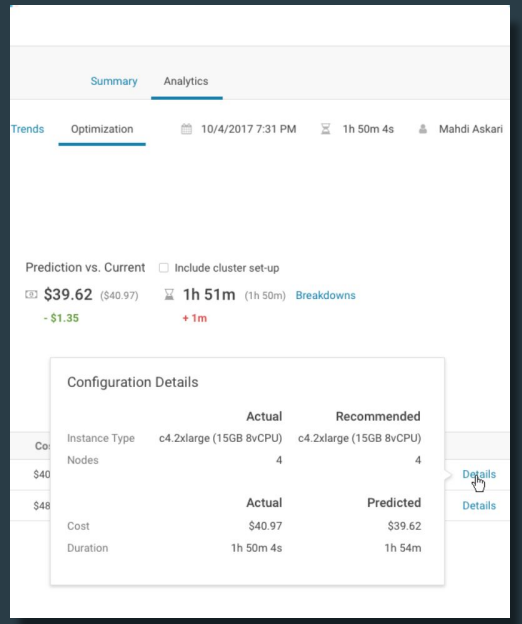
Proactively optimize Workloads, Application Performance, and Infrastructure Capacity for Data Warehousing, Data Engineering, and Machine Learning Environments



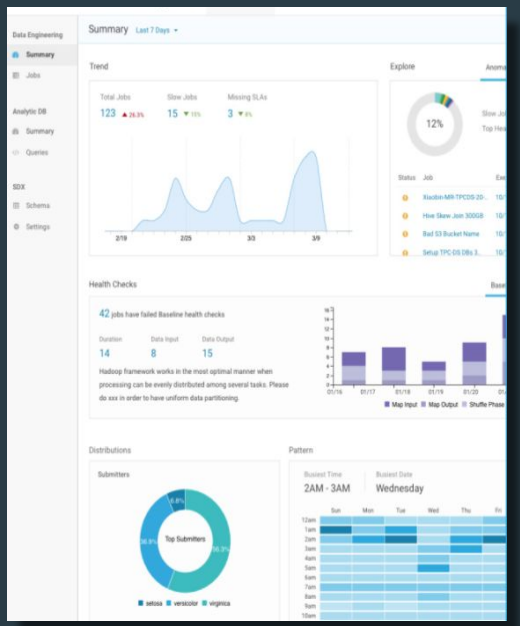
Migrate



Analyze



Optimize



Manage

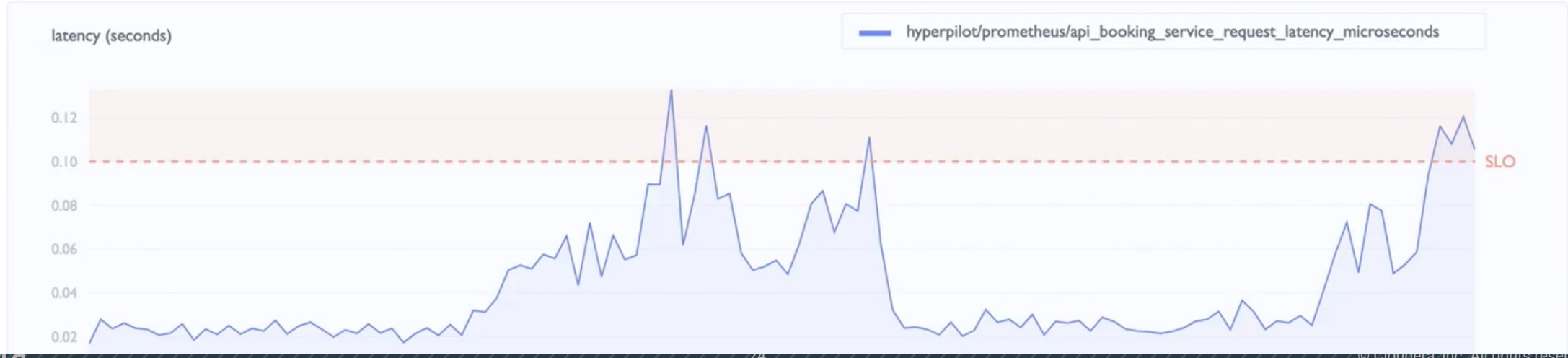
AIOPS ON K8S - NODE BOTTLENECK ANALYSIS

Diagnosis Result of SLO Violation Incident

Time: Jan 16, 2018 4:05 PM

Top Related Problems:

Rank	Description	Score
1	Node Resource Bottleneck: Resource: cpu Node: gke-tech-demo-00dd30ea-default-pool-dbd68a14-5jlb	8.46
2	Node Resource Bottleneck: Resource: cpu Node: gke-tech-demo-00dd30ea-default-pool-dbd68a14-gcx0	8.17
3	Container Over Utilization: Resource: cpu Node: gke-tech-demo-00dd30ea-default-pool-dbd68a14-1ppn Container: pathfinder-3705398937-wn26d	-10.86



AIOPS ON K8S - CLUSTER AUTOMATION

Turn on HyperPilot

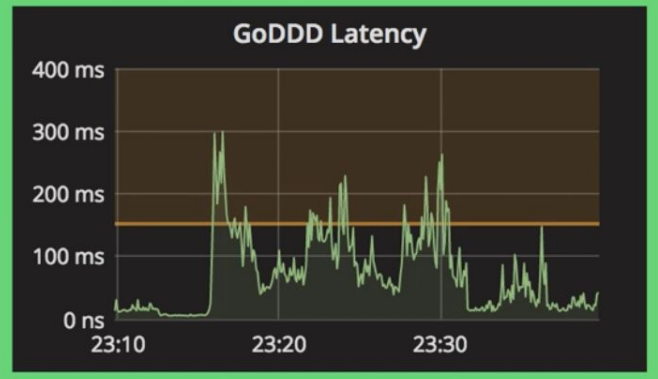
With HyperPilot (HP):

- HP understands network congestion is bottleneck
- Throttles Spark resources
- Restores Web App QoS

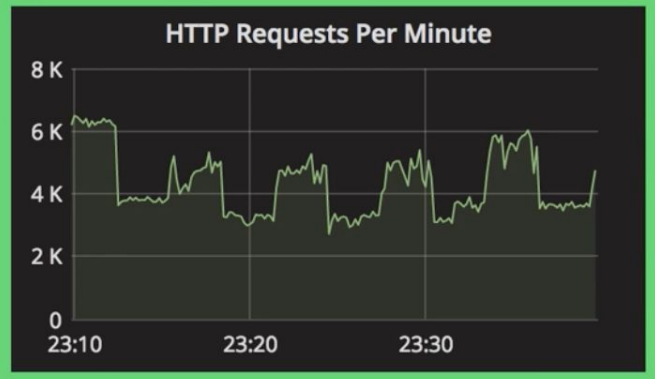
Next Page

High Priority Web Application

Latency | Application QoS

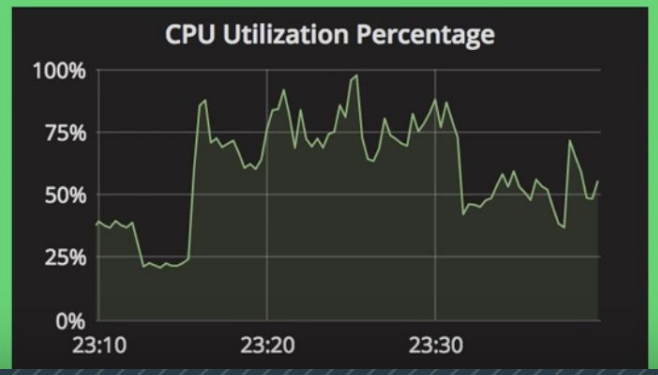


Throughput | Application QoS



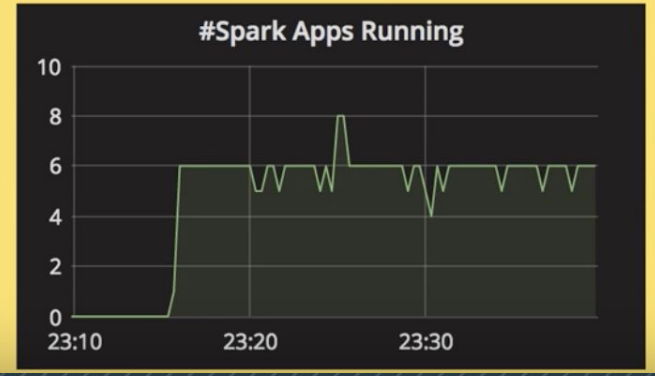
Cluster Resources

Resource Utilization



Low Priority Spark Jobs

Throughput | Application QoS



Relevant Sigs

sig-big-data, sig-ml, sig-scheduling

Want to play around with this stuff?

tiny.cloudera.com/**CML**

Questions?

THANK YOU

cloudera