Apache Hadoop YARN: State of the union

Vinod Kumar Vavilapalli (Apache Hadoop VP, Co-founder of YARN project)

Sunil Govindan (Apache Hadoop PMC)

Speakers

Vinod Kumar Vavilapalli

- Apache Hadoop VP, ASF Member
- Yahoo! -> Hortonworks
- 10 years (only) of Hadoop
- ’Rewritten’ the Hadoop processing side – Became Apache Hadoop YARN
- Running compute platform team at Hortonworks: YARN, MapReduce, Slider, container cloud on YARN

Sunil Govindan

- Apache Hadoop PMC
- Contributing to YARN Scheduler improvements, Integrating TensorFlow to YARN etc
- Staff Engineer @ Hortonworks YARN Team

HORTONWORKS

Agenda

- Introduction
- Past
- State of Union

HORTONWORKS

A brief timeline from past year: GA Releases

- Application Priority
- Reservations
- Node labels improvements

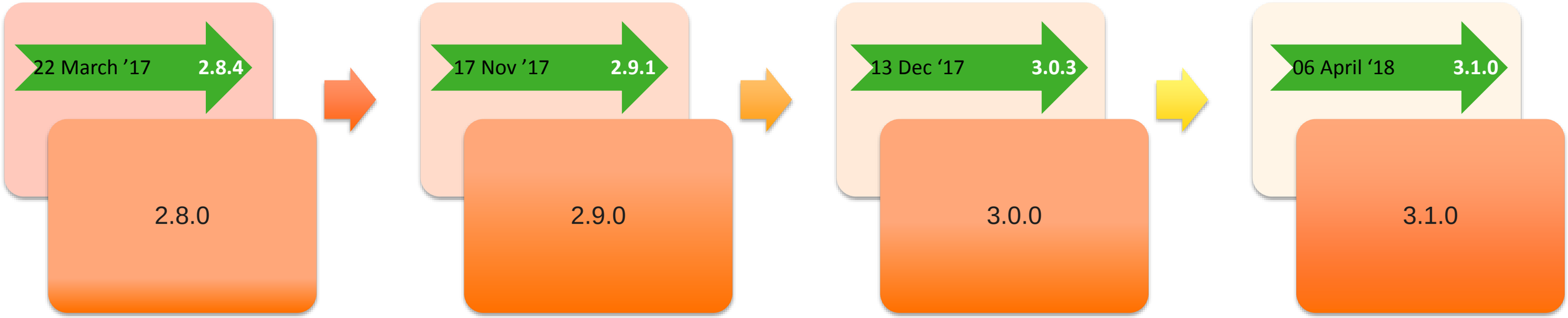
- YARN Federation
- Opportunistic Container

(Backported from 3.0)

- New YARN UI
- Timeline V2

- Global Scheduling
- Multiple Resource types
- New YARN UI
- Timeline service V2

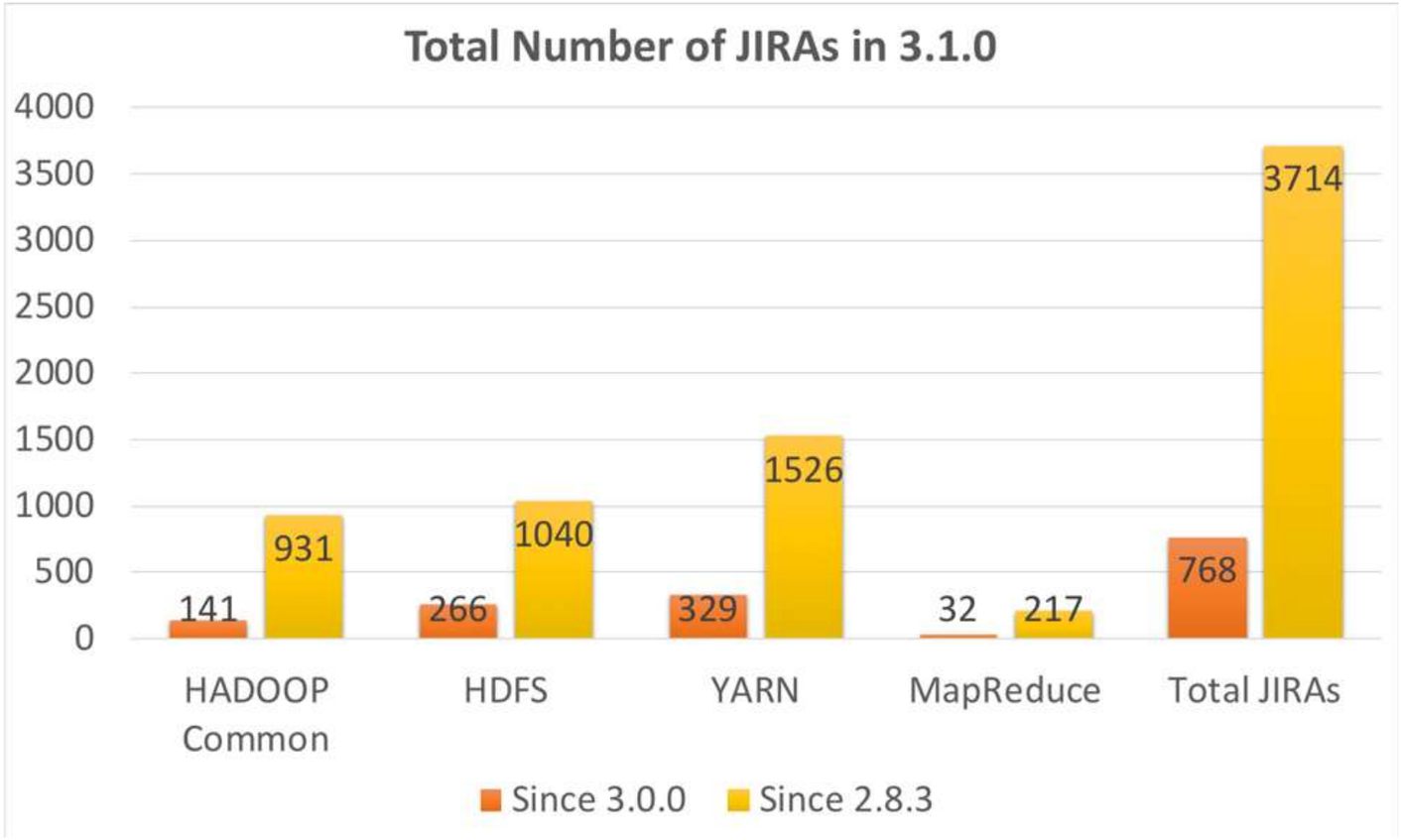
- GPU/FPGA
- YARN Native Service
- Placement Constraints



Ever involving requirements (computation intensive, larger, services)

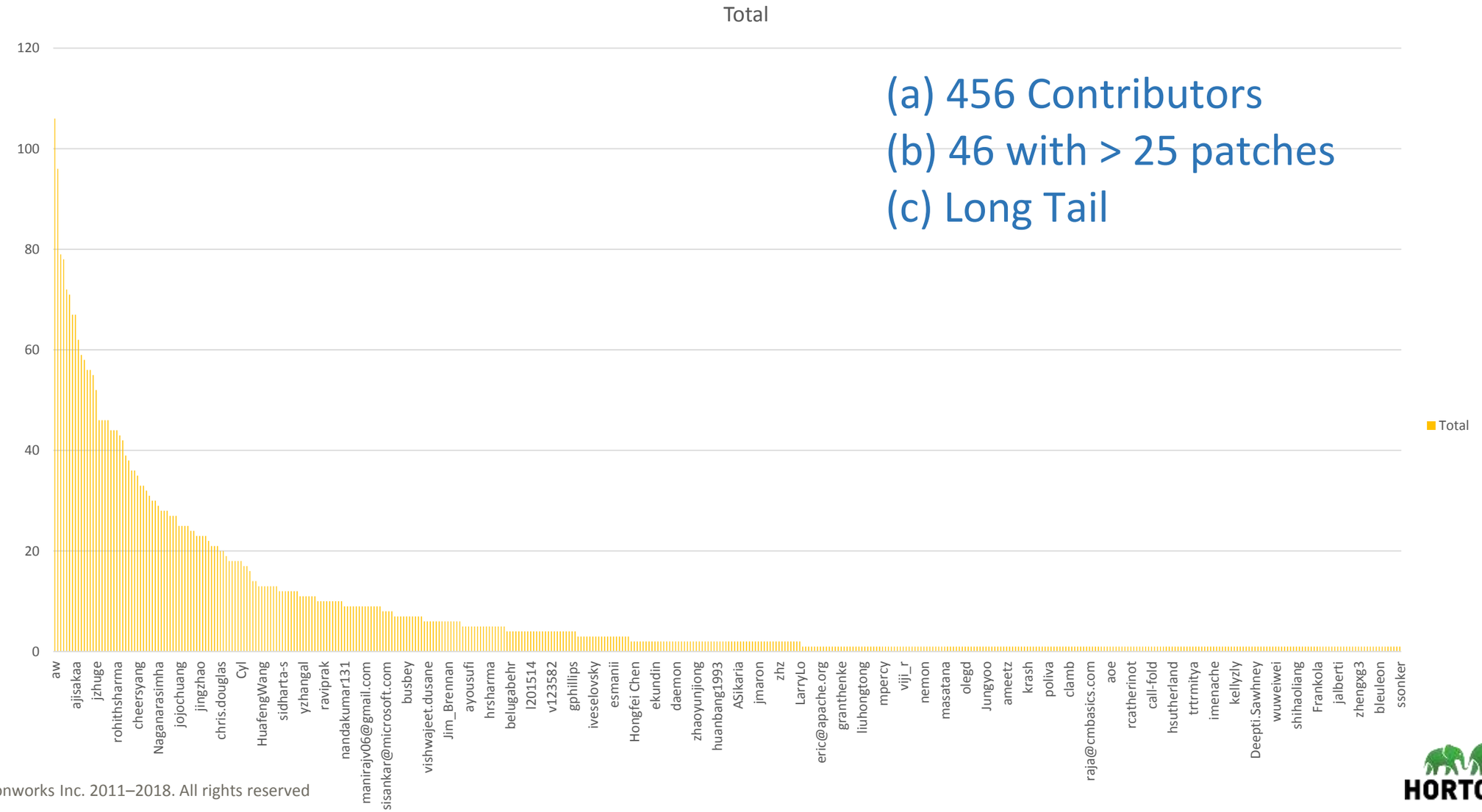
HORTONWORKS

Community Update: JIRAs in 3.1.0

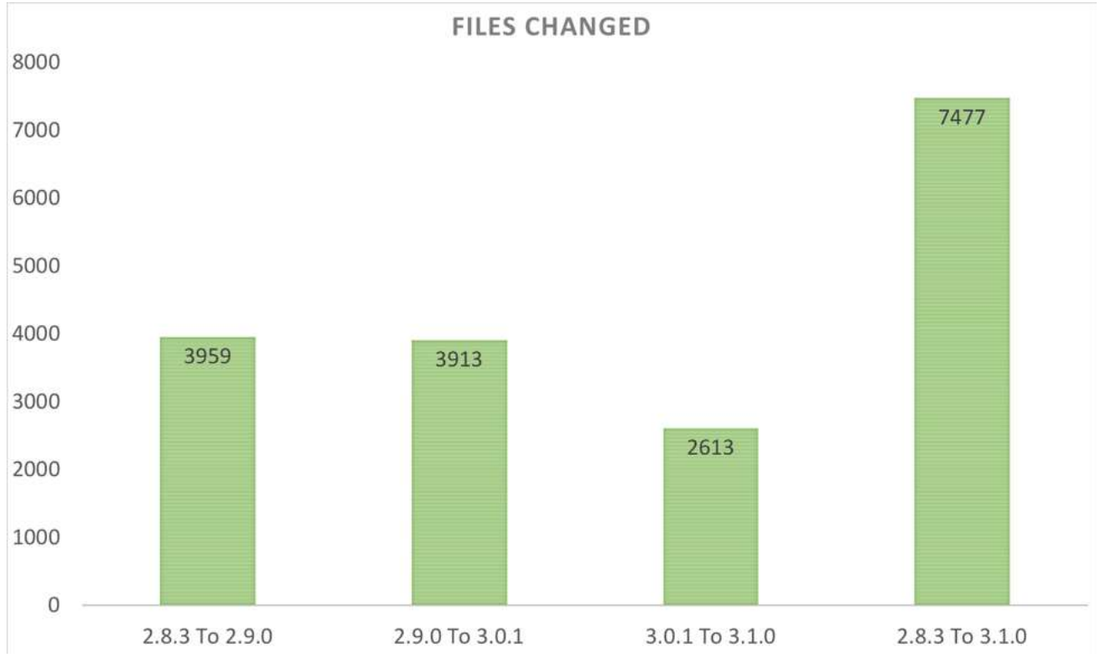


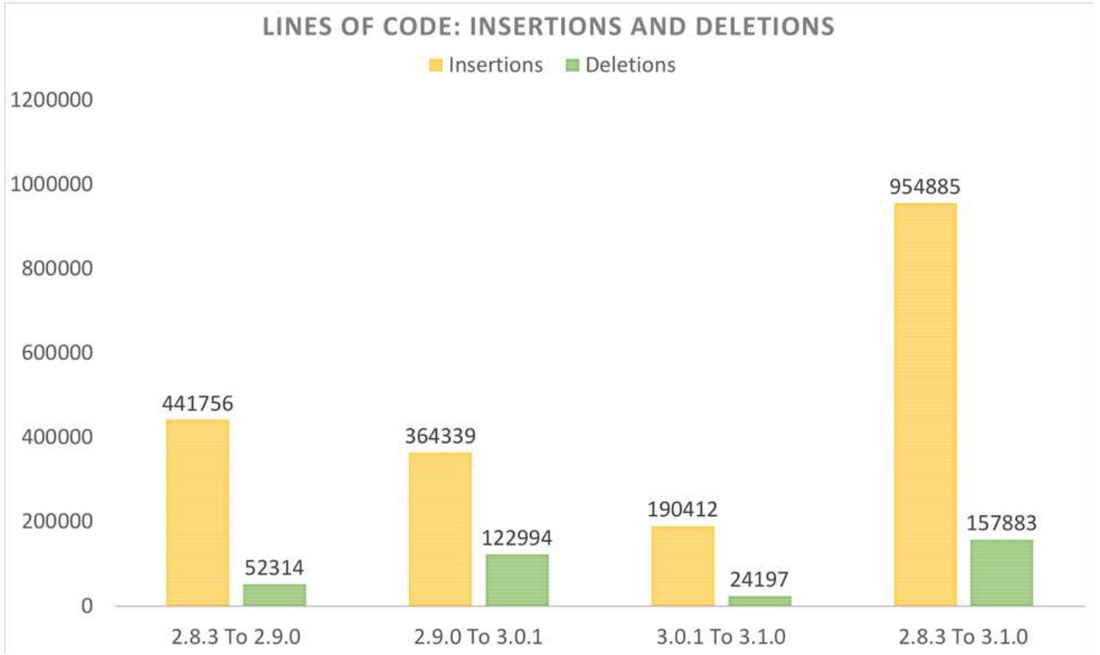
HORTONWORKS

Community Update: Contributors 2.8.3 -> 3.1.0



Community Update: Source code changes

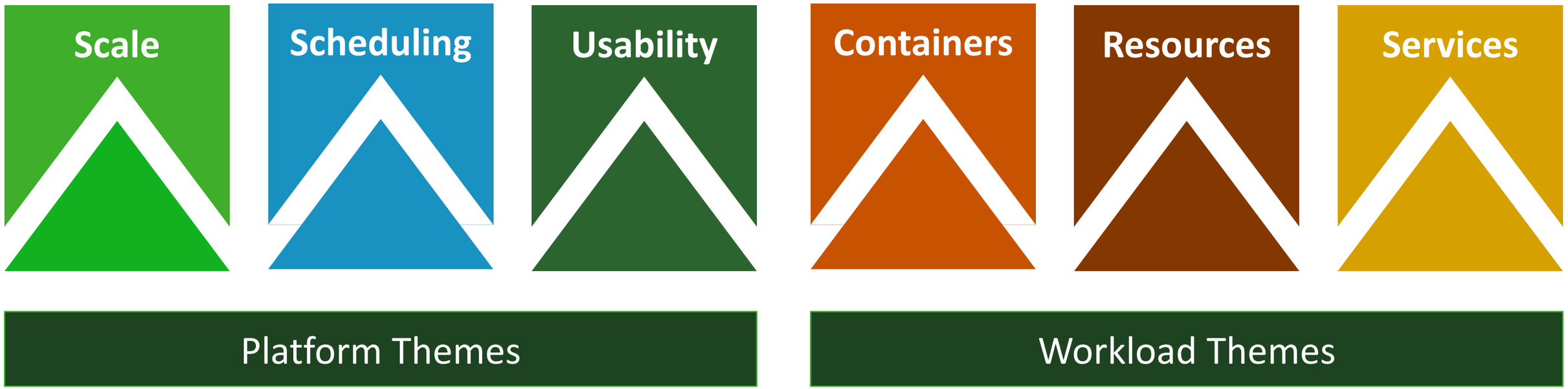




HORTONWORKS

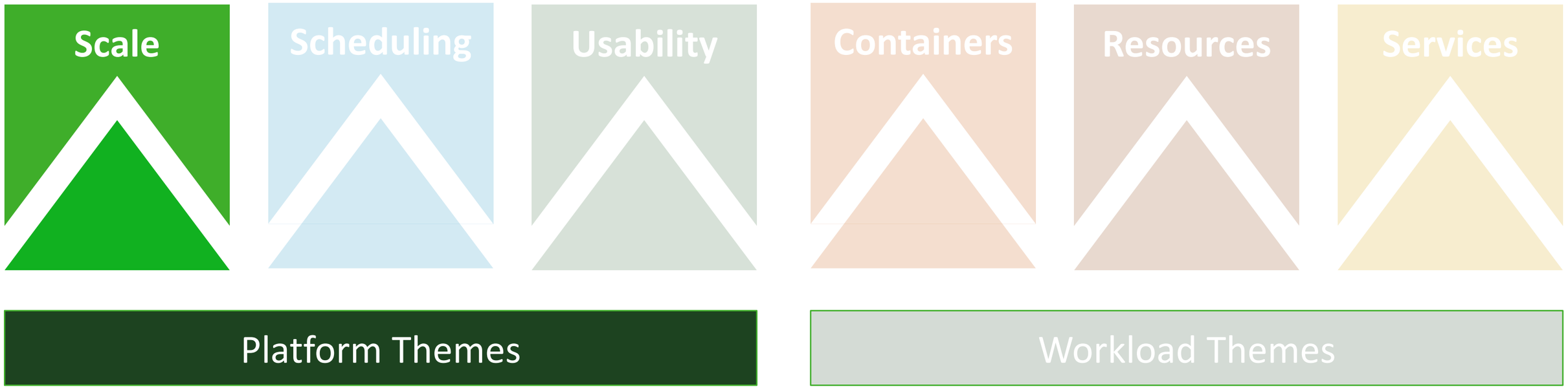

Apache Hadoop 3.0/3.1
HORTONWORKS

Key Themes



HORTONWORKS

Key Themes



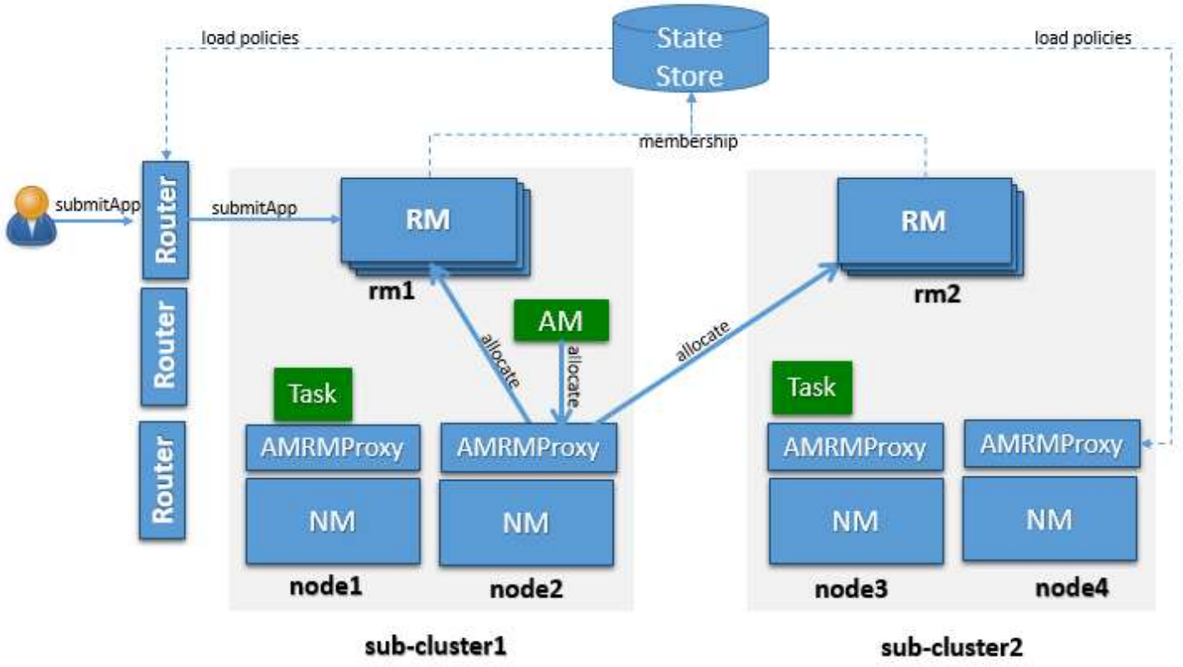
Looking at the Scale!

- Tons of sites with clusters made up of large amount of nodes
  - Oath (Yahoo!), Twitter, LinkedIn, Microsoft, Alibaba etc.
- Before last year Summit, largest clusters
  - 6K-8K
- Now: 40K nodes (federated), 20K nodes (single cluster).
- Roadmap: To 100K and beyond

HORTONWORKS

YARN Federation

- Enables applications to scale to **100k** of thousands of nodes
- Federation divides a large (10-100k nodes) cluster into smaller units called sub-clusters
- Federation negotiates with sub-clusters RM's and provide resources to the application
- Applications can schedule tasks on any node



HORTONWORKS

-- Related Session --

YARN federation: taming a beasty fleet with global optimizations

Carlo Curino (Microsoft) & Subru Krishnan (Microsoft)

https://dataworkssummit.com/san-jose-2018/session/federation-v2-global-optimizations/

Thursday, June 21 9:30 AM – 10:10 AM, Executive Ballroom 210D/H

HORTONWORKS

-- Related Session --

Apache Hadoop YARN 3.x in Alibaba

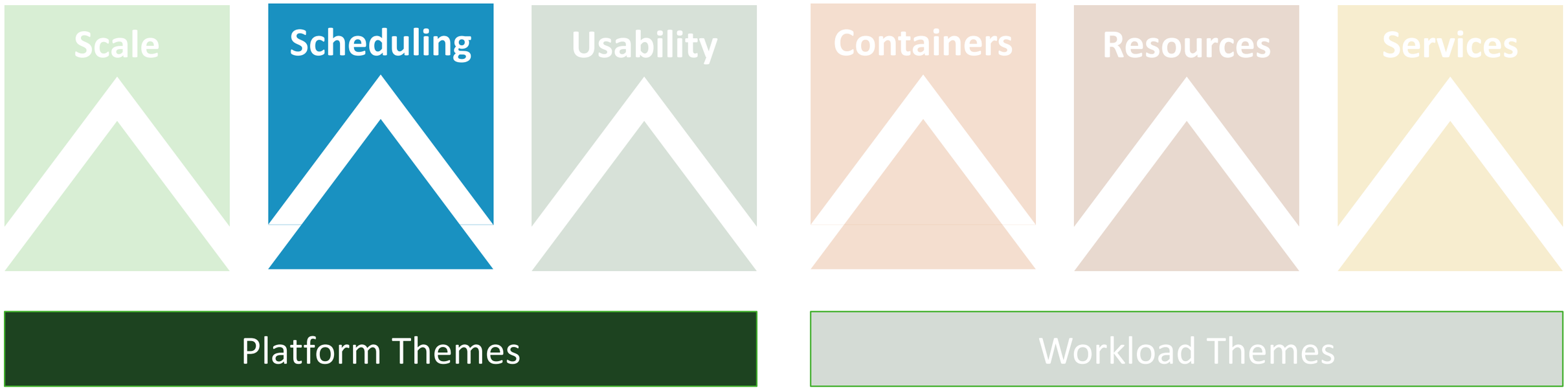
Weiwei Yang (Alibaba Group) & Ren Chunde (Alibaba Group)

https://dataworkssummit.com/san-jose-2018/session/apache-hadoop-yarn-3-x-in-alibaba/

Thursday, June 21 10:20 AM – 11:00 AM, Meeting Room 211A/B/C/D

HORTONWORKS

Key Themes



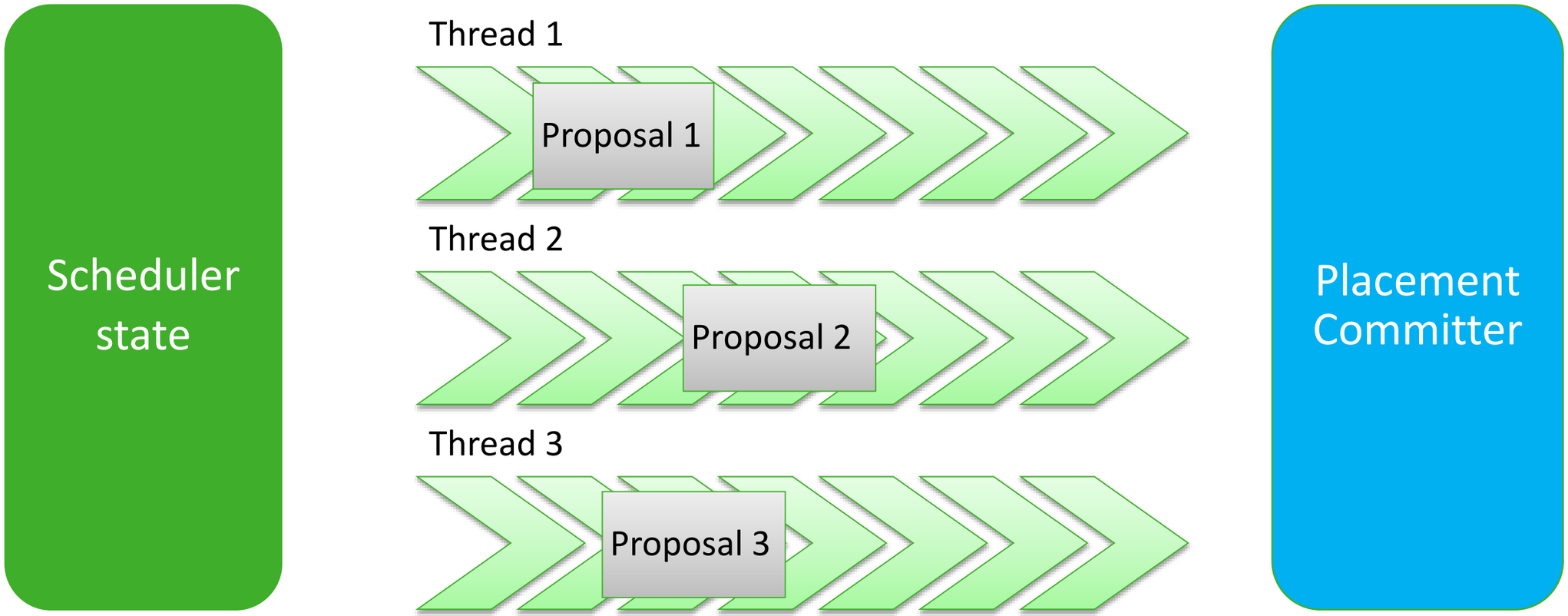
HORTONWORKS

Moving towards Global & Fast Scheduling

- **YARN-5139**
- **Problems**
  - Current design of one-node-at-a-time allocation cycle can lead to suboptimal decisions.
  - Several coarse grained locks.
- **With this, we improved to**
  - Look at several nodes at a time
  - Fine grained locks
  - Multiple allocator threads
  - YARN scheduler can allocate 3k+ containers per second ≈ 10 mil allocations / hour!
  - 10X throughput gains
  - Much better placement decisions

HORTONWORKS

Global Scheduling explained



Better placement strategies (YARN-6592)

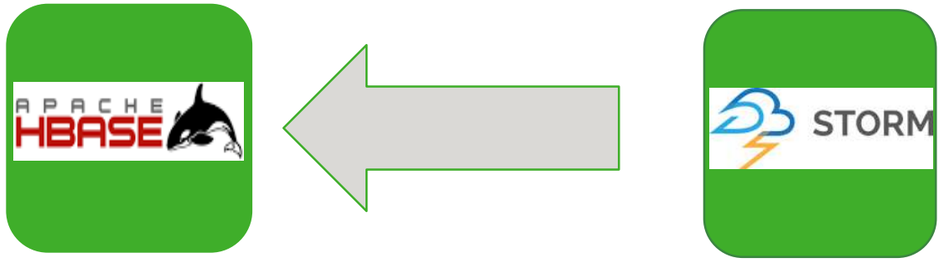
- **Past**
  - Supported constraints in form of Node Locality

- **Now YARN can support a lot more use cases**
  - Co-locate the allocations of a job on the same rack (**affinity**)
  - Spread allocations across machines (**anti-affinity**) to minimize resource interference
  - Allow up to a specific number of allocations in a node group (**cardinality**)

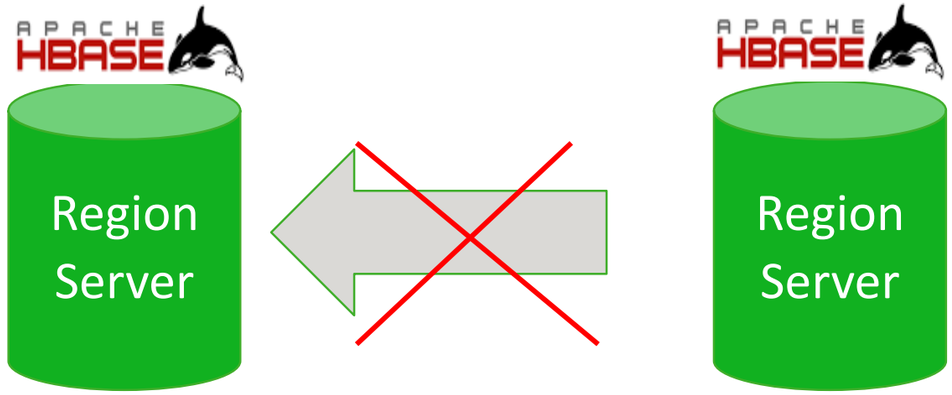
HORTONWORKS

Better placement strategies (YARN-6592)

- Affinity



- Anti-affinity



HORTONWORKS

-- Related Session --

Rich placement constraints: Who said YARN cannot schedule services?

Konstantinos Karanasos (Microsoft) and Wangda Tan (Hortonworks)

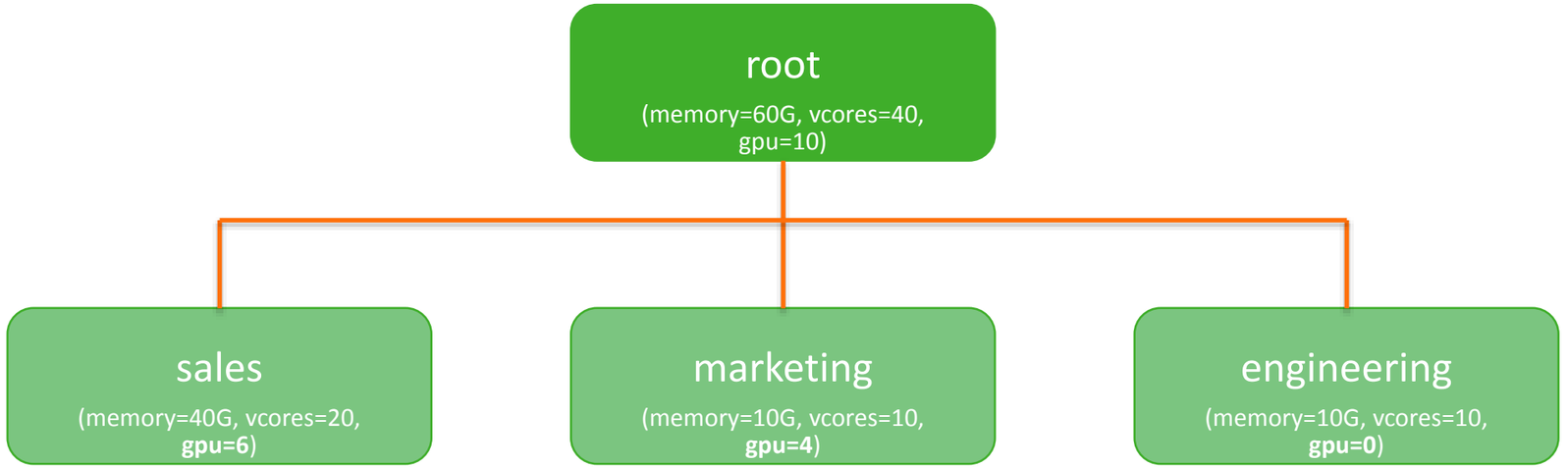
https://dataworkssummit.com/san-jose-2018/session/rich-placement-constraints-who-said-yarn-cannot-schedule-services/

Wednesday, June 20 11:50 AM – 12:30 PM, Executive Ballroom 210C/G

HORTONWORKS

Absolute Resources Configuration in CS – YARN-5881

- The cloud model! “Give me X resources, not X%”
- Gives ability to configure Queue resources as below
  <memory=24GB, vcores=20, yarn.io/gpu=2>
- Enables admins to assign different quotas of different resource-types
- No more **“Single percentage value limitation for all resource-types”**



HORTONWORKS

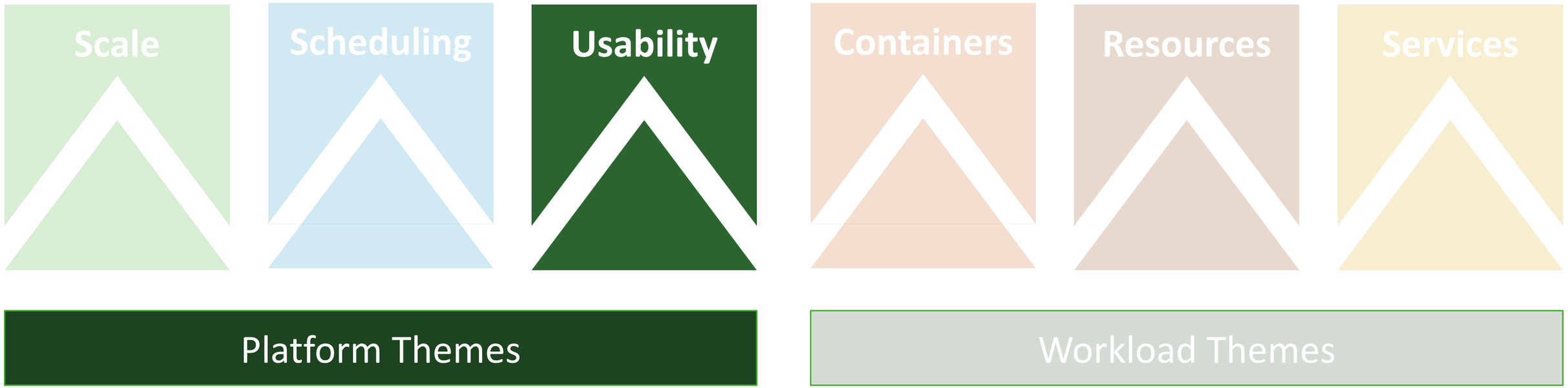
Auto Creation of Leaf Queues - YARN-7117

- **Easily map a queue** explicitly to user or group with out additional configs
- For e.g, User X comes in, automatically create a queue for user X with a templated capacity requirements
- **Auto created Queues** will be
  - created runtime based on user mapping
  - cleaned up after use
  - adhering to ACLs

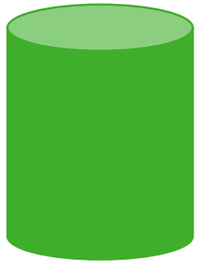
Reservation-based Scheduling: If You're Late Don't Blame Us! - Carlo, et al. 2015

HORTONWORKS

Key Themes



Usability: Queue & Logs

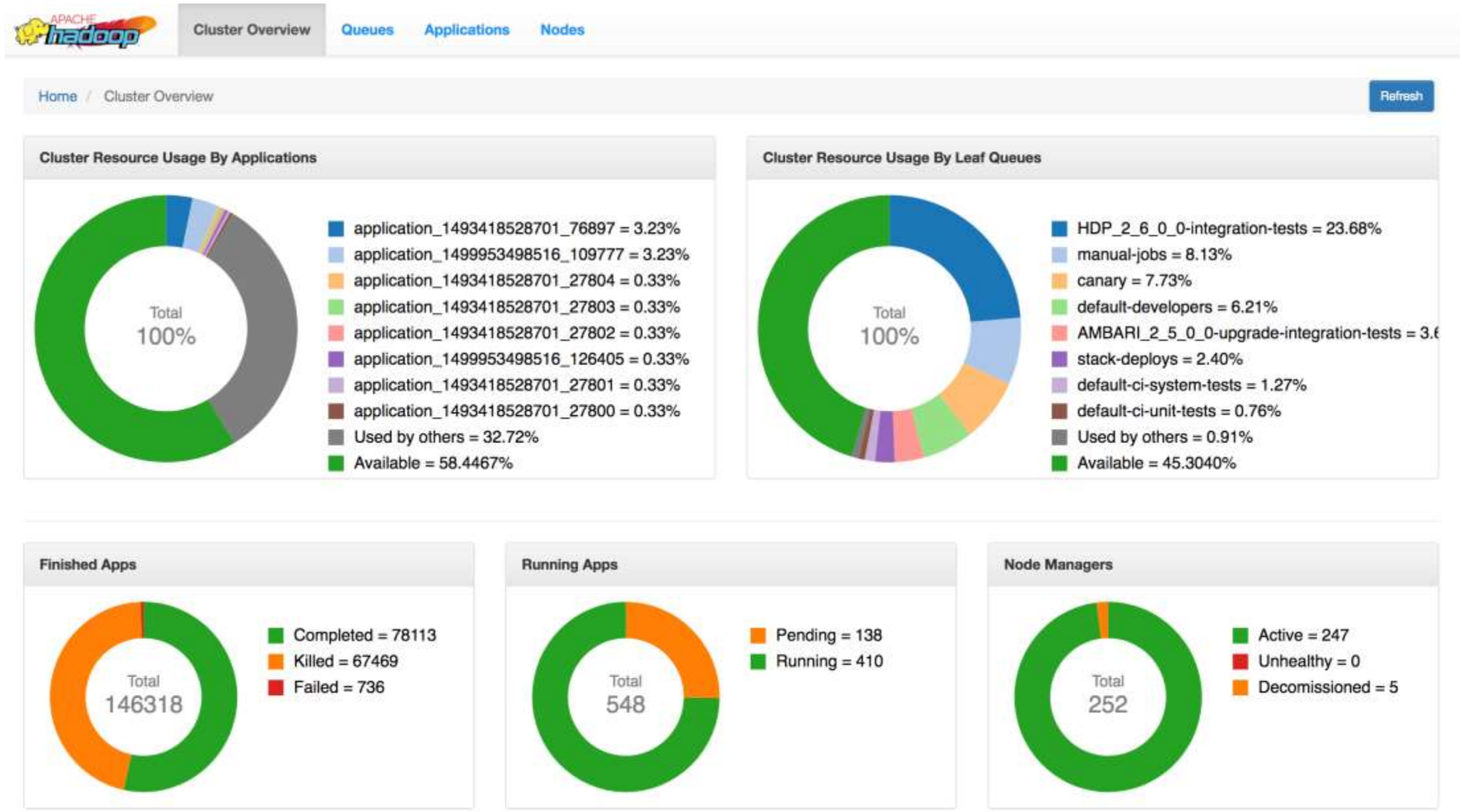


Improved logs management
(YARN-4904)
Live application logs

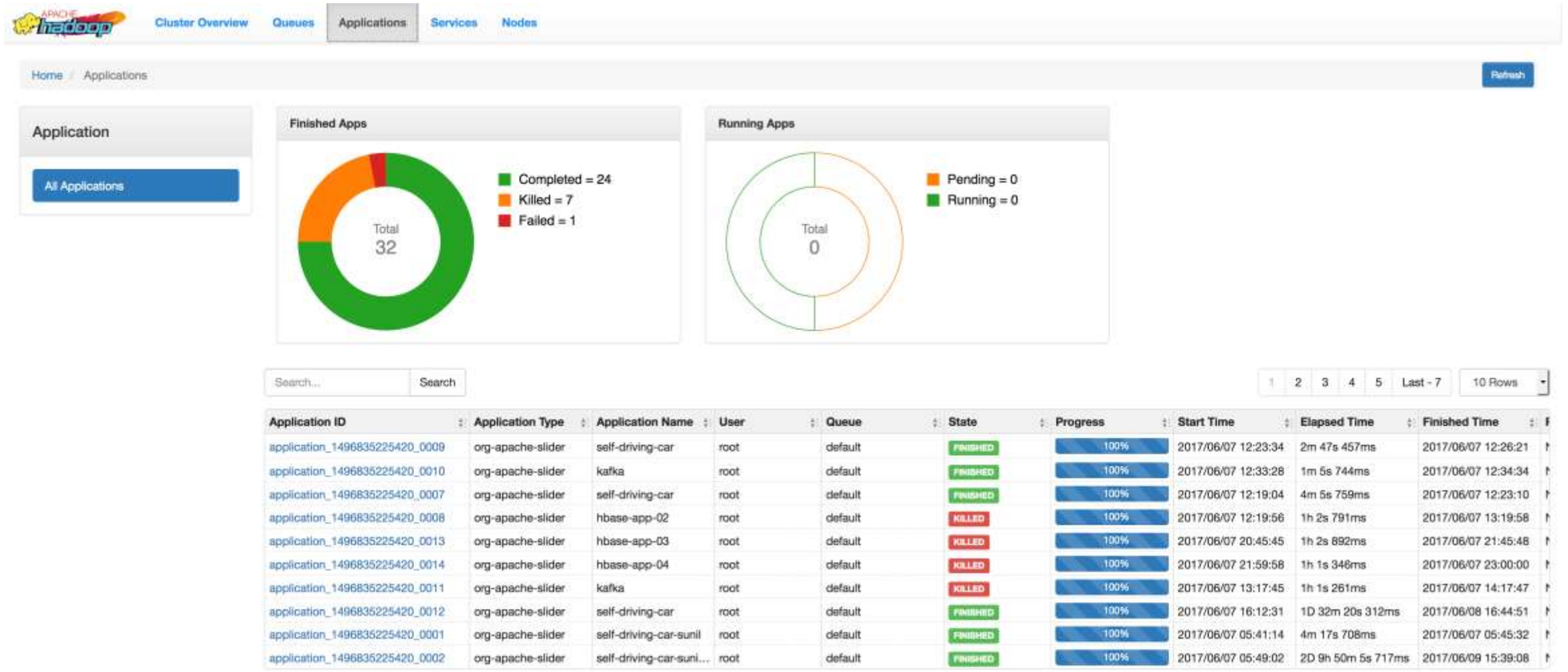
API based queue management
Decentralized
(YARN-5734)

HORTONWORKS

Usability: UI 1/2



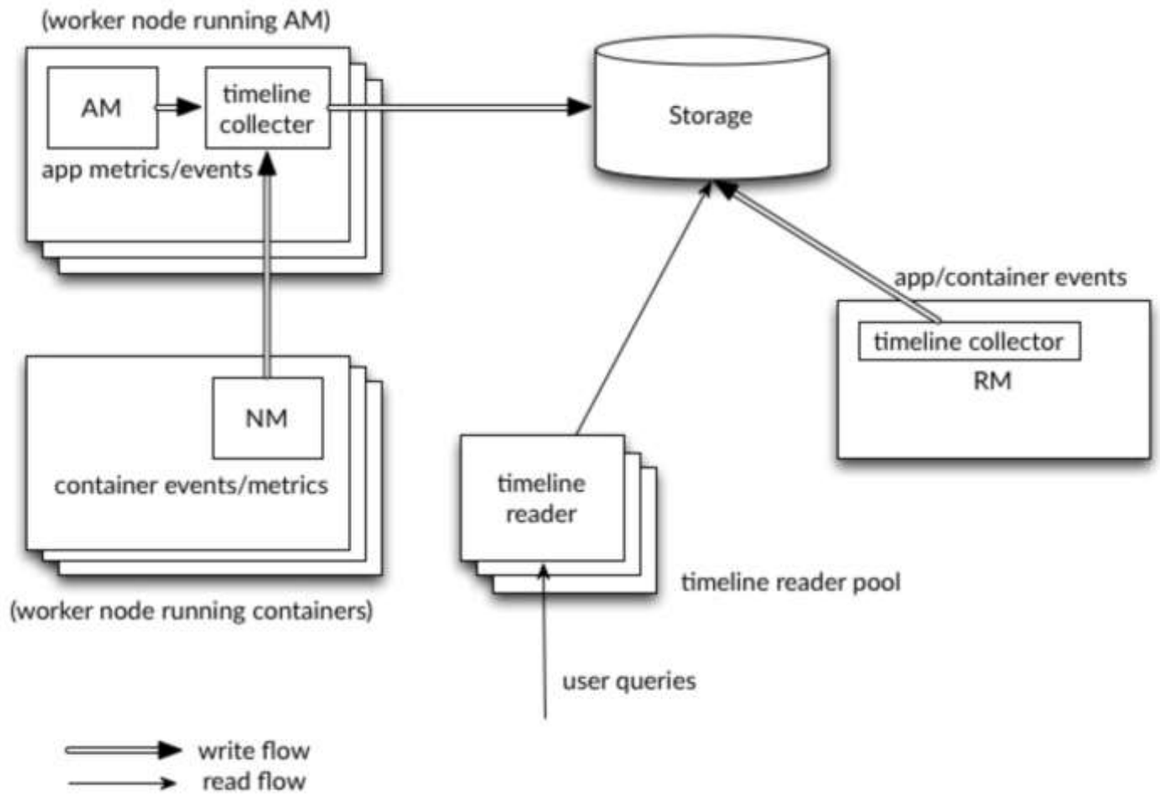
Usability: UI 2/2



HORTONWORKS

Timeline Service 2.0

- Understanding and Monitoring a Hadoop cluster itself is a BigData problem
  - Using **HBase as backend** for better scalability for read/write
  - More robust storage fault tolerance
  - Migration and compatibility with v.1.5



-- Related Session --

Migrating your clusters and workloads from Hadoop 2 to Hadoop 3

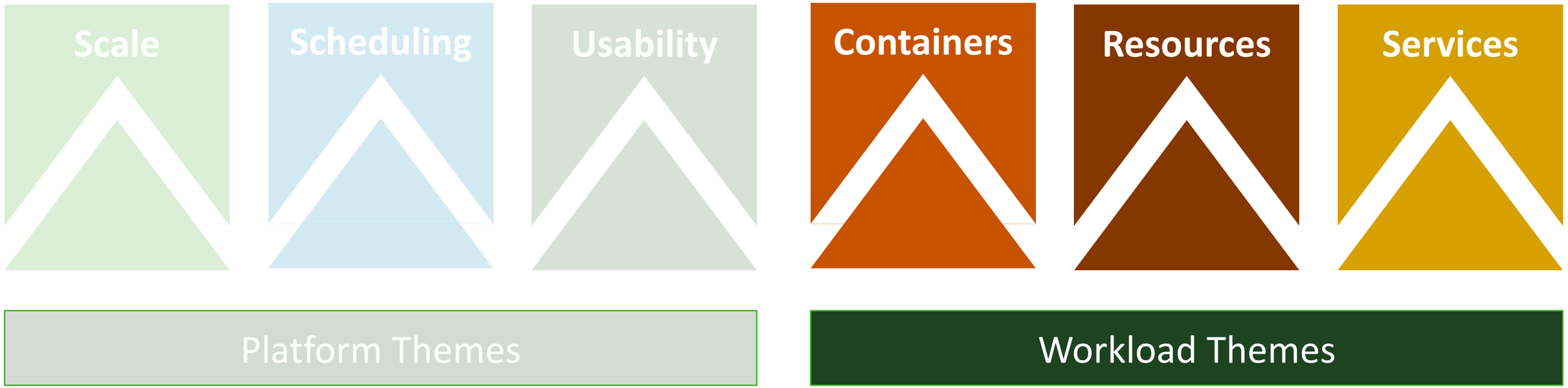
Suma Shivaprasad (Hortonworks) & Rohith Sharma (Hortonworks)

https://dataworkssummit.com/san-jose-2018/session/ease-of-migration-from-hadoop-2-to-hadoop-3-clusters/

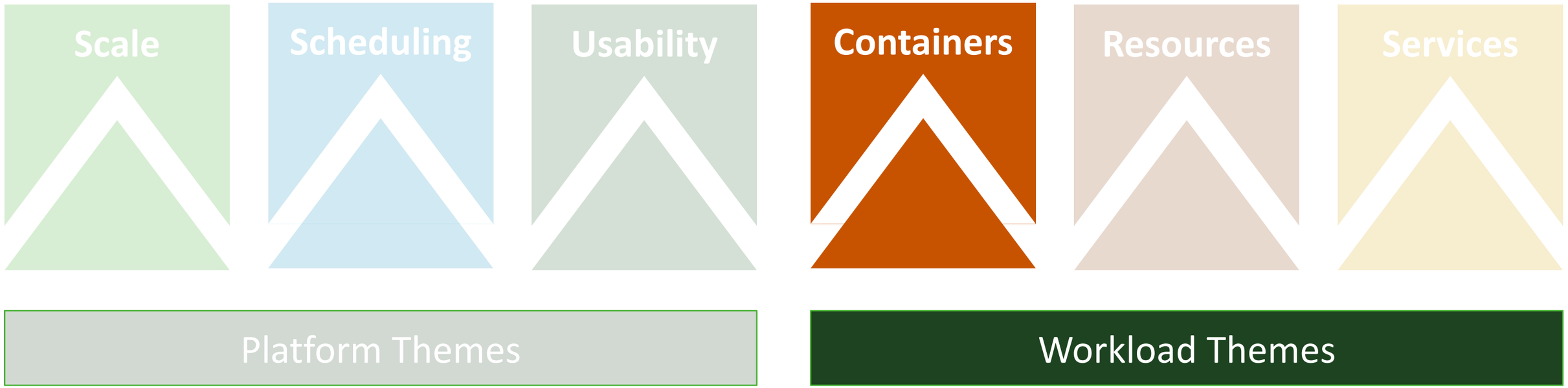
Tuesday, June 19 2:50 PM – 3:30 PM, Meeting Room 211A/B/C/D

HORTONWORKS

Key Themes

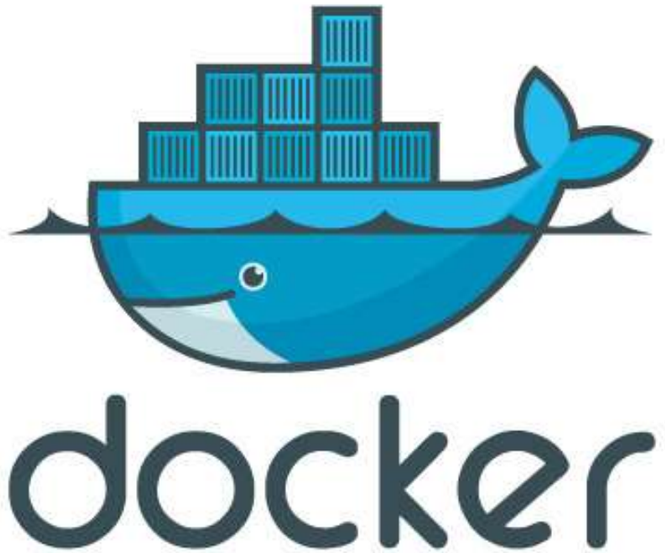


Key Themes



Containers

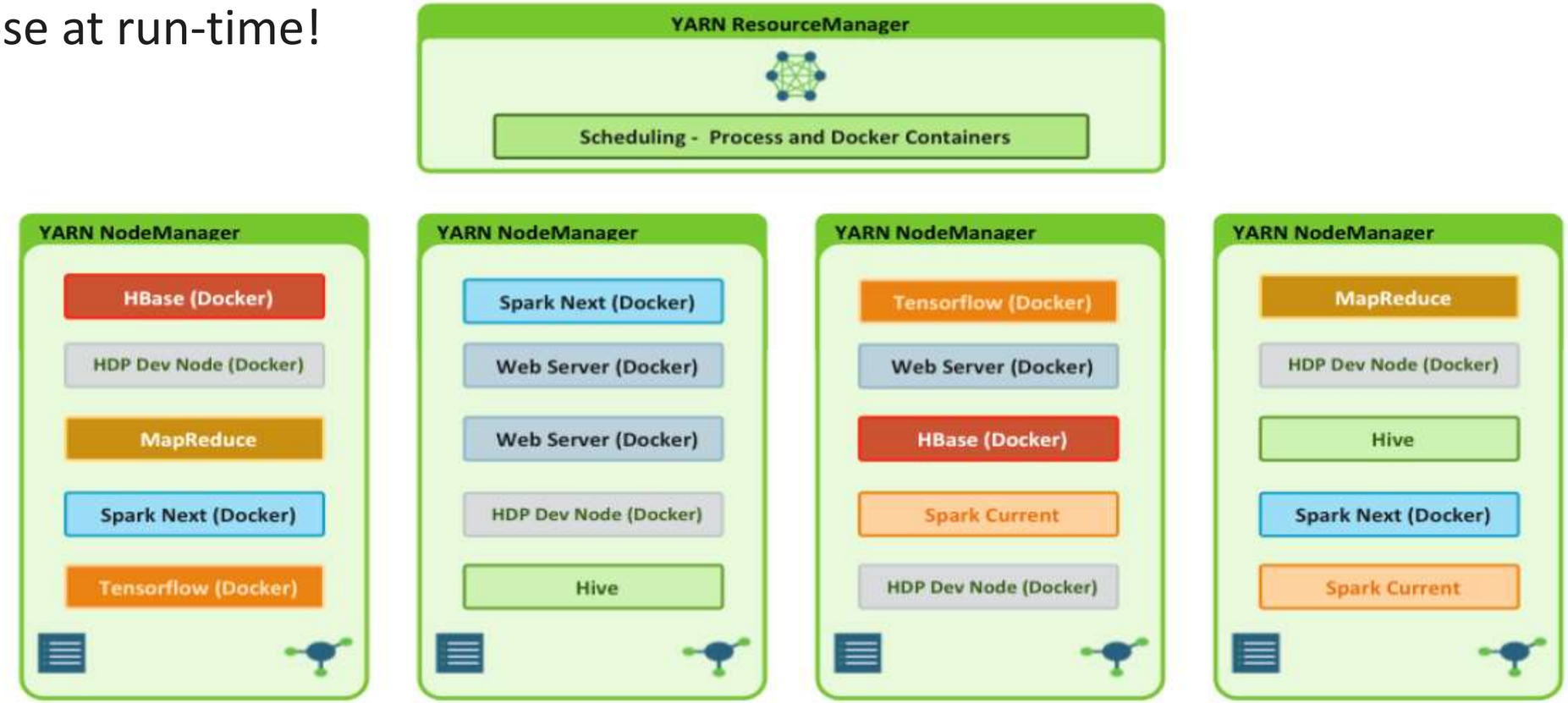
- Better Packaging model
  - Lightweight mechanism for packaging and resource isolation
  - Popularized and made accessible by Docker
- Native integration ++ in YARN
  - Support for “Container Runtimes” in LCE: YARN-3611
  - Process runtime
  - **Docker runtime**



HORTONWORKS

Containers

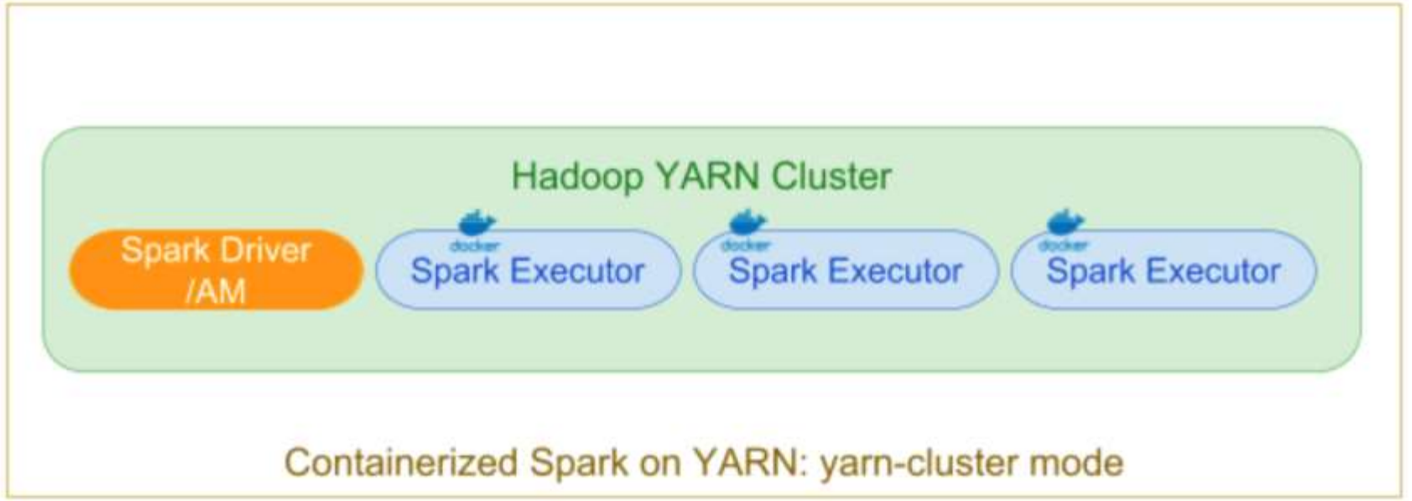
- Run both with and without docker on the same cluster
- Choose at run-time!



HORTONWORKS

Spark on Docker in YARN

- **Apache Spark** applications have a complex set of required software dependencies
- Docker on YARN helps to solve Spark **package isolation issues with**
  - **PySpark** - Python versions, packages
  - **R** - packages



HORTONWORKS

What about running all of Big Data containerized?

- YARN – Big Data apps, moving to generic apps with containerization
- K8S – industry standard orchestration layer for generic apps
- We have done YARN on YARN! Next slide
- YARN on K8S? K8S on YARN? Run them side-by-side?
- What does containerized BigData mean?
  - Lift and Shift?
  - Break up every service?

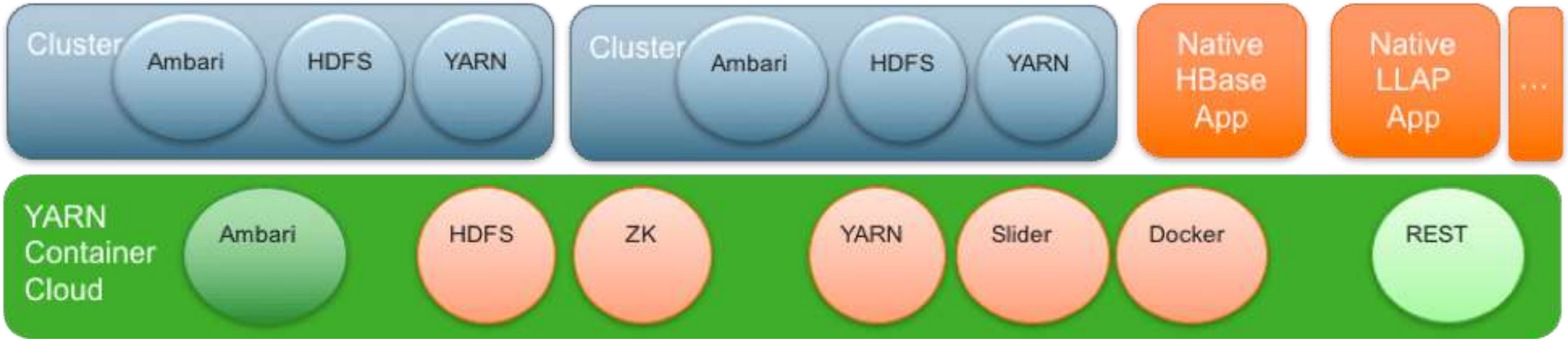
HORTONWORKS

Ycloud: YARN Based Container Cloud

Running Hadoop clusters on Hadoop (YARN)

- ✓ Each system-test cluster is an app on YARN!
- ✓ Each system-test cluster is a group of containers
- ✓ YInception: YARN on YARN ☺

✓ *Testing Hadoop on Hadoop!*



HORTONWORKS

-- Related Session --

Containers and Big Data

Billie Rinaldi (Hortonworks) and Shane Kumpf (Hortonworks)

https://dataworkssummit.com/san-jose-2018/session/containers-and-big-data/

Wednesday, June 20 2:00 PM – 2:40 PM, Grand Ballroom 220A

HORTONWORKS

-- Related Session --

Quality for the Hadoop Zoo

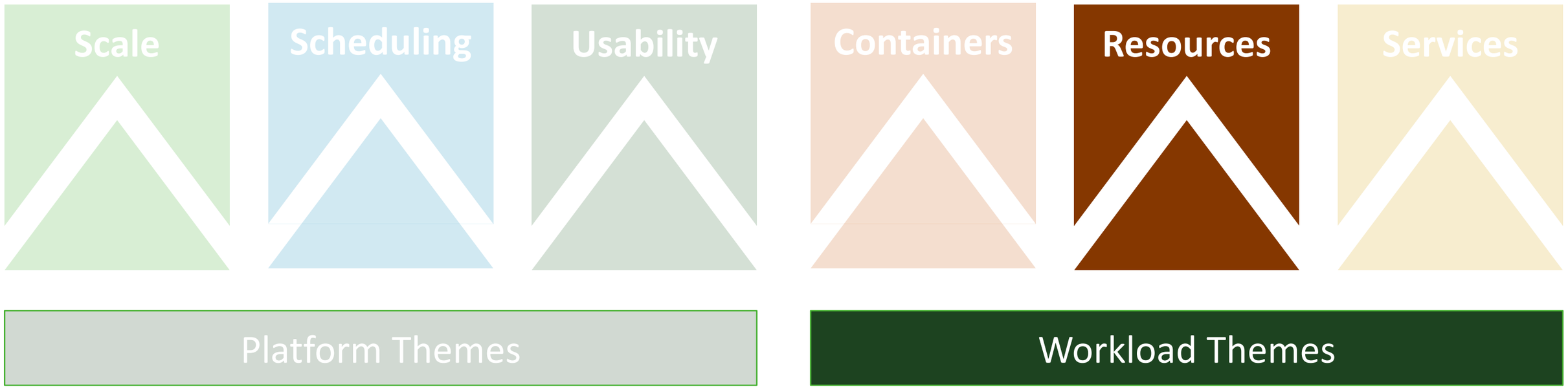
Sunitha Velpula (Hortonworks)

https://dataworkssummit.com/san-jose-2018/session/quality-for-the-hadoop-zoo/

Thursday, June 21 12:20 PM - 1:00 PM Meeting Room 230B

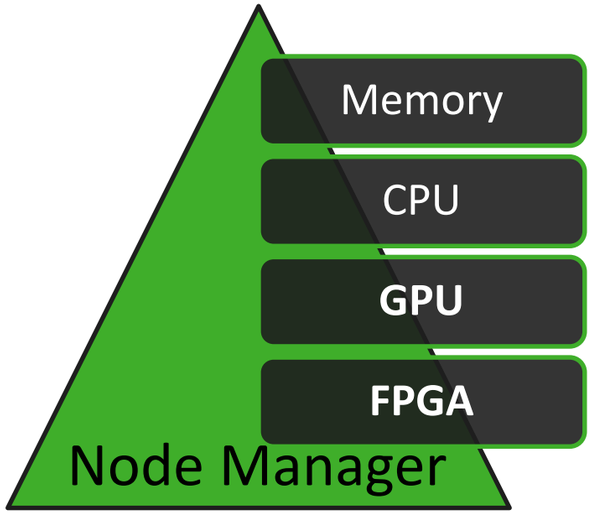
HORTONWORKS

Key Themes



Resource profiles and custom resource types

- **YARN** supported only **Memory** and **CPU**
- **Now**
  - A generalized vector for all resources
  - Admin could add arbitrary resource types!



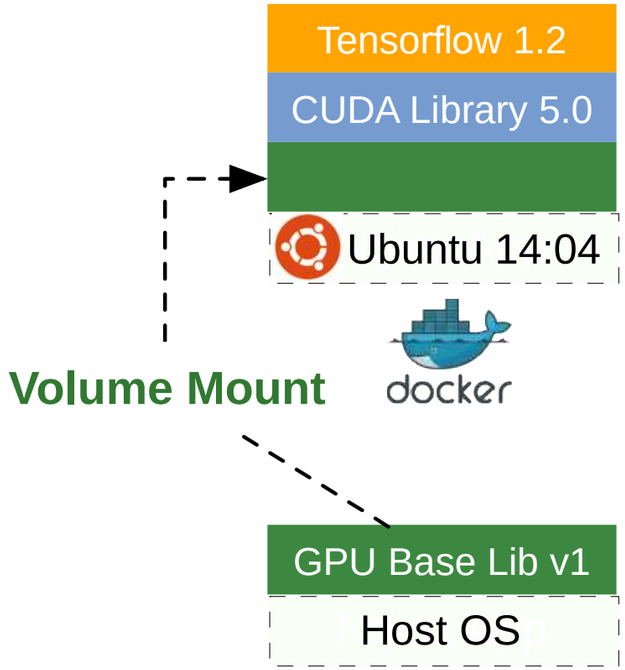
- Ease of resource requesting model using profiles for apps

Profile	Memory	CPU	GPU
Small	2 GB	4 Cores	0 Cores
Medium	4 GB	8 Cores	0 Cores
Large	16 GB	16 Cores	4 Cores

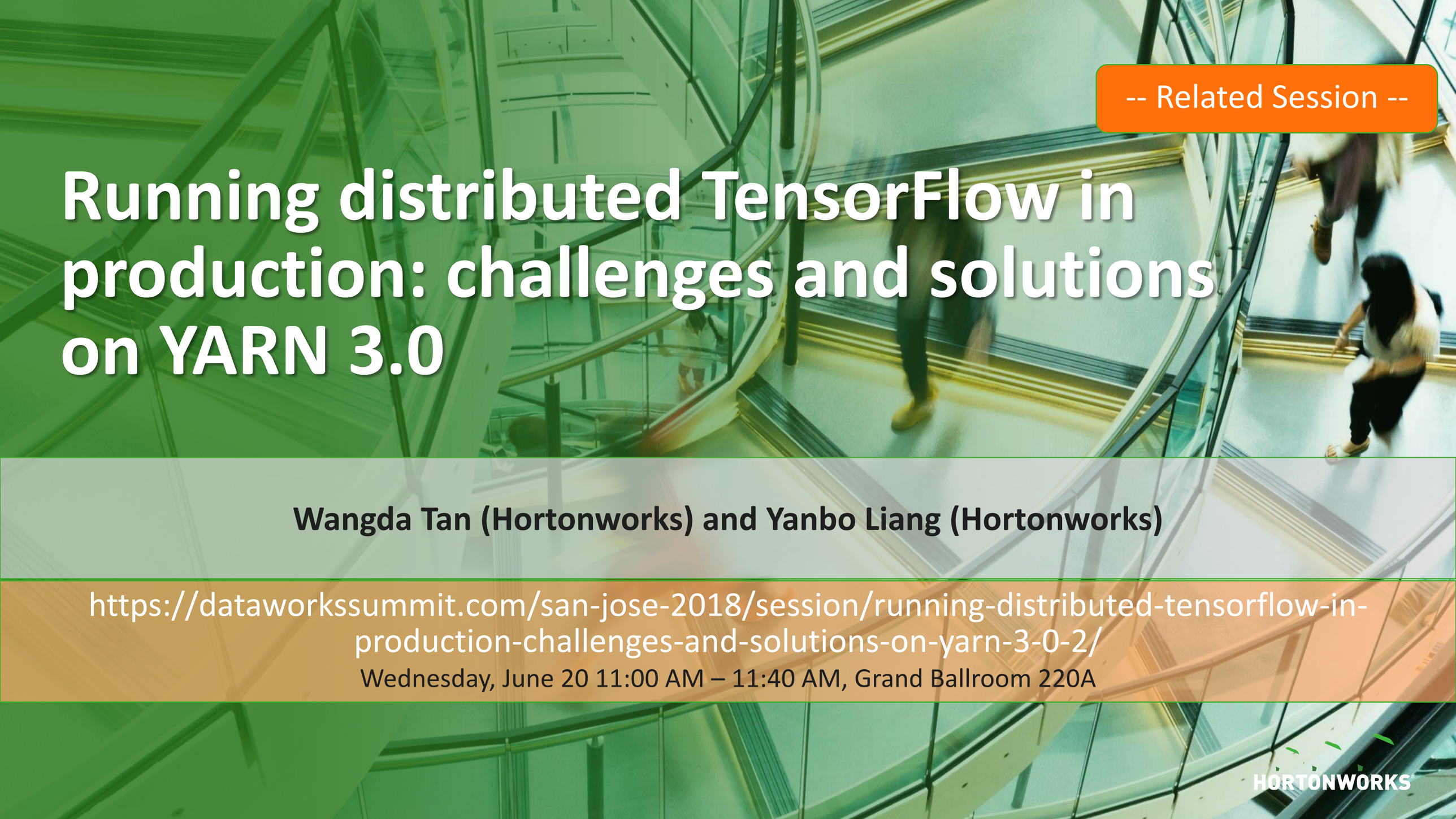
HORTONWORKS

GPU support on YARN

- **Why?**
  - No need to setup separate clusters
  - Leverage shared compute!
- **Why need isolation?**
  - Multiple processes use the single GPU will be:
    - Serialized.
    - Cause OOM easily.
- **GPU isolation on YARN:**
  - Granularity is for per-GPU device.
  - Use cgroups / docker to enforce isolation.



HORTONWORKS



-- Related Session --

Running distributed TensorFlow in production: challenges and solutions on YARN 3.0

Wangda Tan (Hortonworks) and Yanbo Liang (Hortonworks)

https://dataworkssummit.com/san-jose-2018/session/running-distributed-tensorflow-in-production-challenges-and-solutions-on-yarn-3-0-2/

Wednesday, June 20 11:00 AM – 11:40 AM, Grand Ballroom 220A

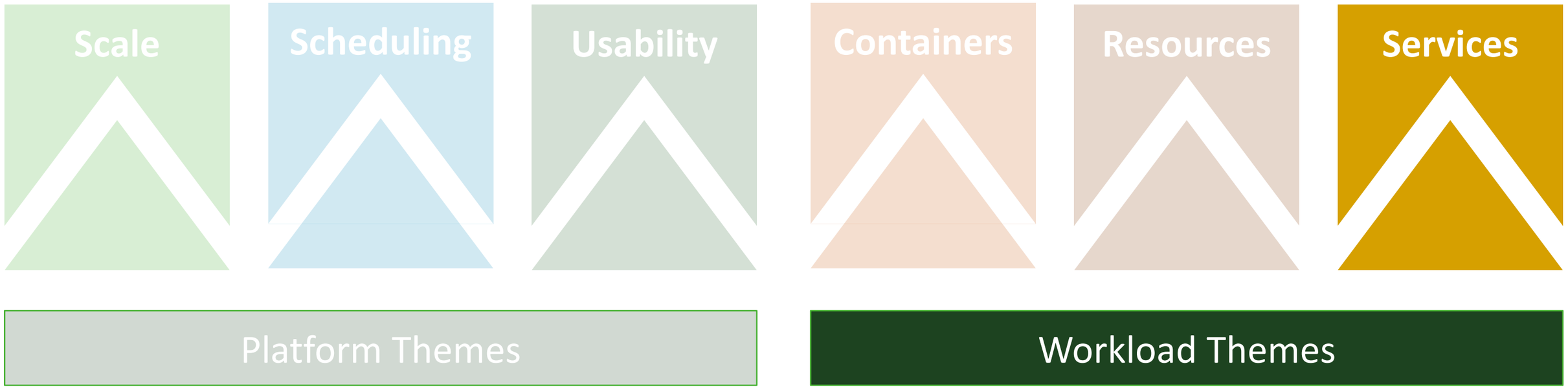
HORTONWORKS

FPGA on YARN

- FPGA isolation on YARN: .
  - Granularity is for per-FPGA device.
  - Use Cgroups to enforce the isolation.
- Currently, only Intel OpenCL SDK for FPGA is supported. But implementation is extensible to other FPGA SDK.

HORTONWORKS

Key Themes



Services support in YARN

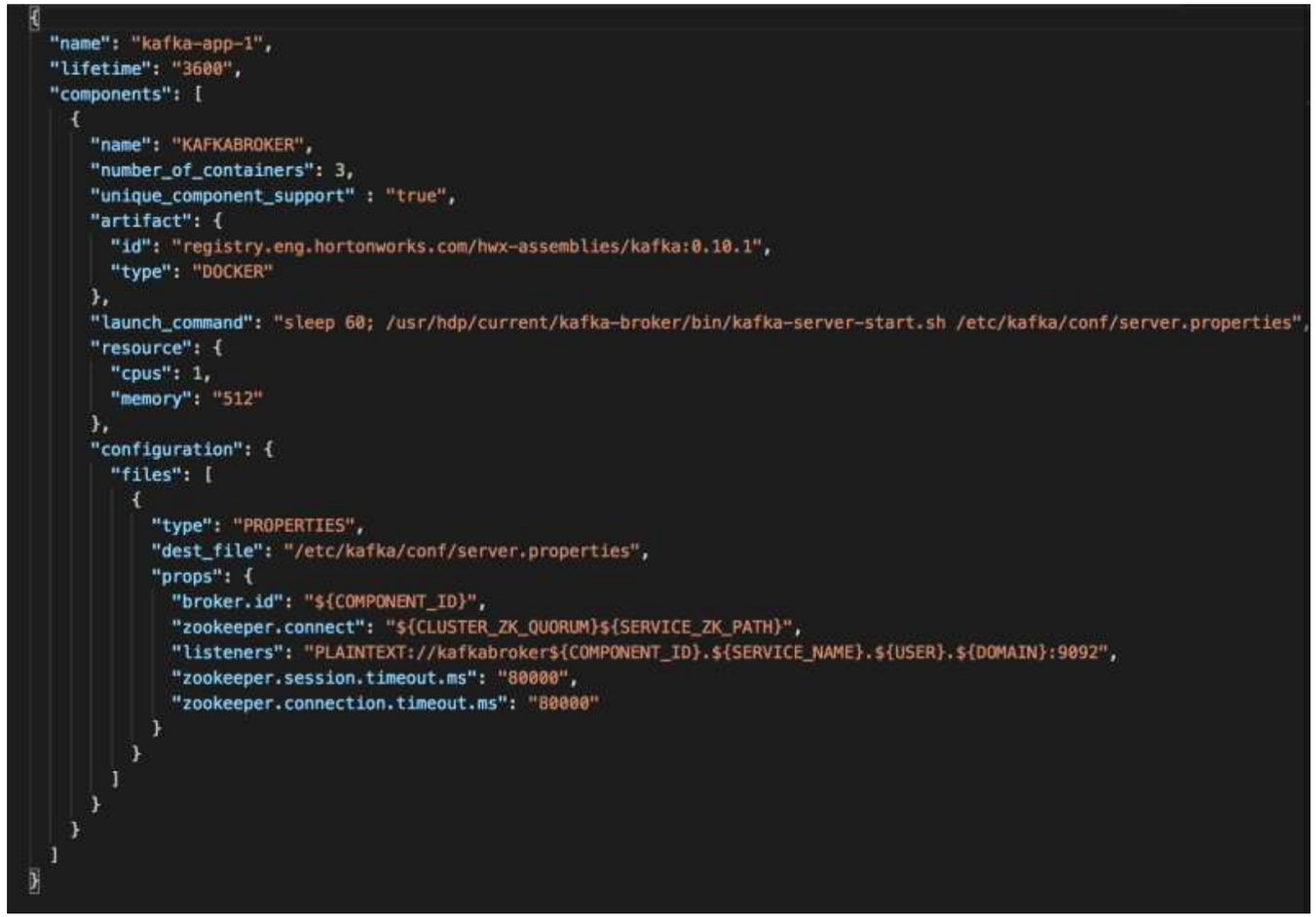
- A native **YARN services framework**
  - YARN-4692
  - [Umbrella] Native YARN framework layer for services and beyond
  - Apache Slider retired from Incubator – lessons and key code carried over to YARN
- Simplified discovery of services via **DNS mechanisms**: YARN-4757
  - ***regionserver-0.hbase-app-3.hadoop.yarn.site***
- **Application & Services upgrades**
  - “Do an upgrade of my HBase app with minimal impact to end-users”
  - YARN-4726

HORTONWORKS

Simplified APIs for service definitions

- Applications need simple APIs
- Need to be deployable “easily”

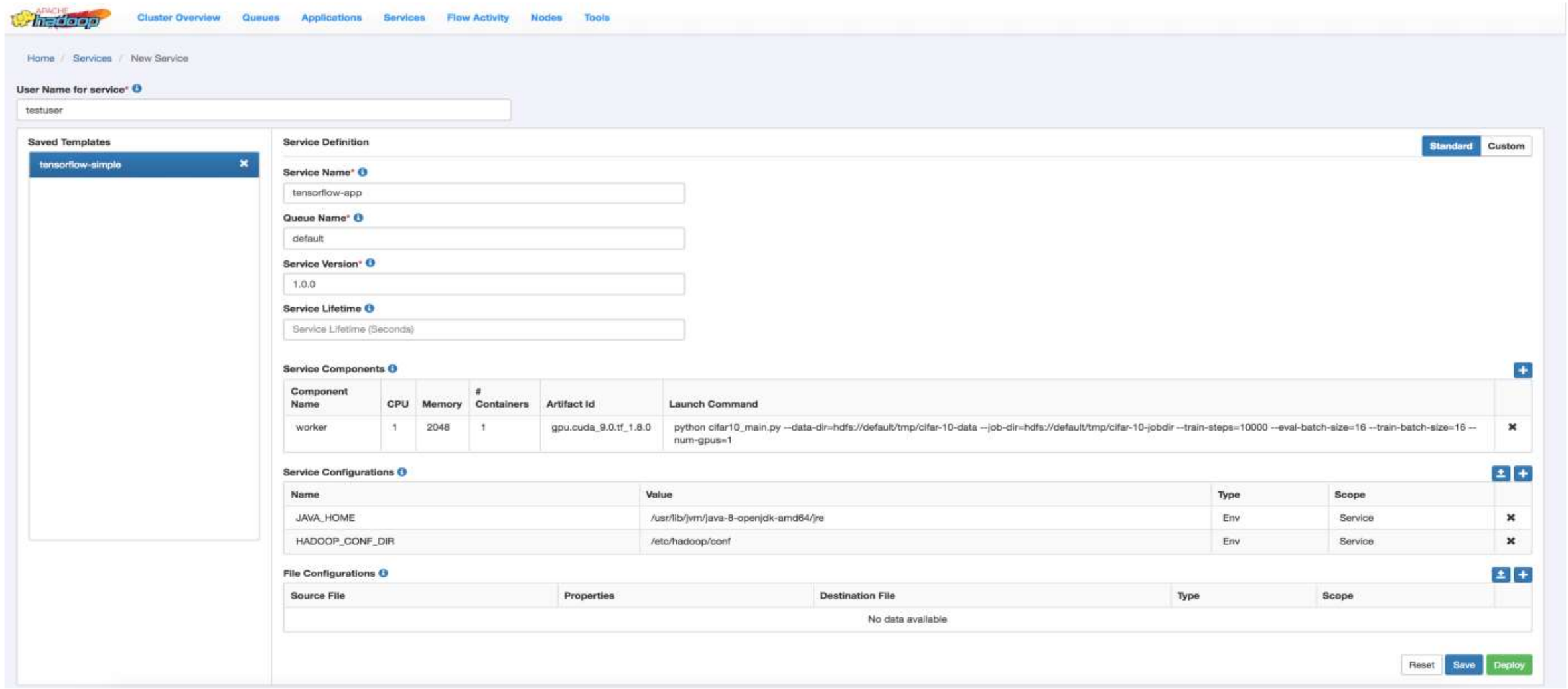
- Simple REST API layer fronting YARN
  - YARN-4793 Simplified API layer
- Spawn services & Manage them



```
{
  "name": "kafka-app-1",
  "lifetime": "3600",
  "components": [
    {
      "name": "KAFKABROKER",
      "number_of_containers": 3,
      "unique_component_support" : "true",
      "artifact": {
        "id": "registry.eng.hortonworks.com/hwx-assemblies/kafka:0.10.1",
        "type": "DOCKER"
      },
      "launch_command": "sleep 60; /usr/hdp/current/kafka-broker/bin/kafka-server-start.sh /etc/kafka/conf/server.properties",
      "resource": {
        "cpus": 1,
        "memory": "512"
      },
      "configuration": {
        "files": [
          {
            "type": "PROPERTIES",
            "dest_file": "/etc/kafka/conf/server.properties",
            "props": {
              "broker.id": "${COMPONENT_ID}",
              "zookeeper.connect": "${CLUSTER_ZK_QUORUM}${SERVICE_ZK_PATH}",
              "listeners": "PLAINTEXT://kafkabroker${COMPONENT_ID}.${SERVICE_NAME}.${USER}.${DOMAIN}:9092",
              "zookeeper.session.timeout.ms": "80000",
              "zookeeper.connection.timeout.ms": "80000"
            }
          }
        ]
      }
    }
  ]
}
```

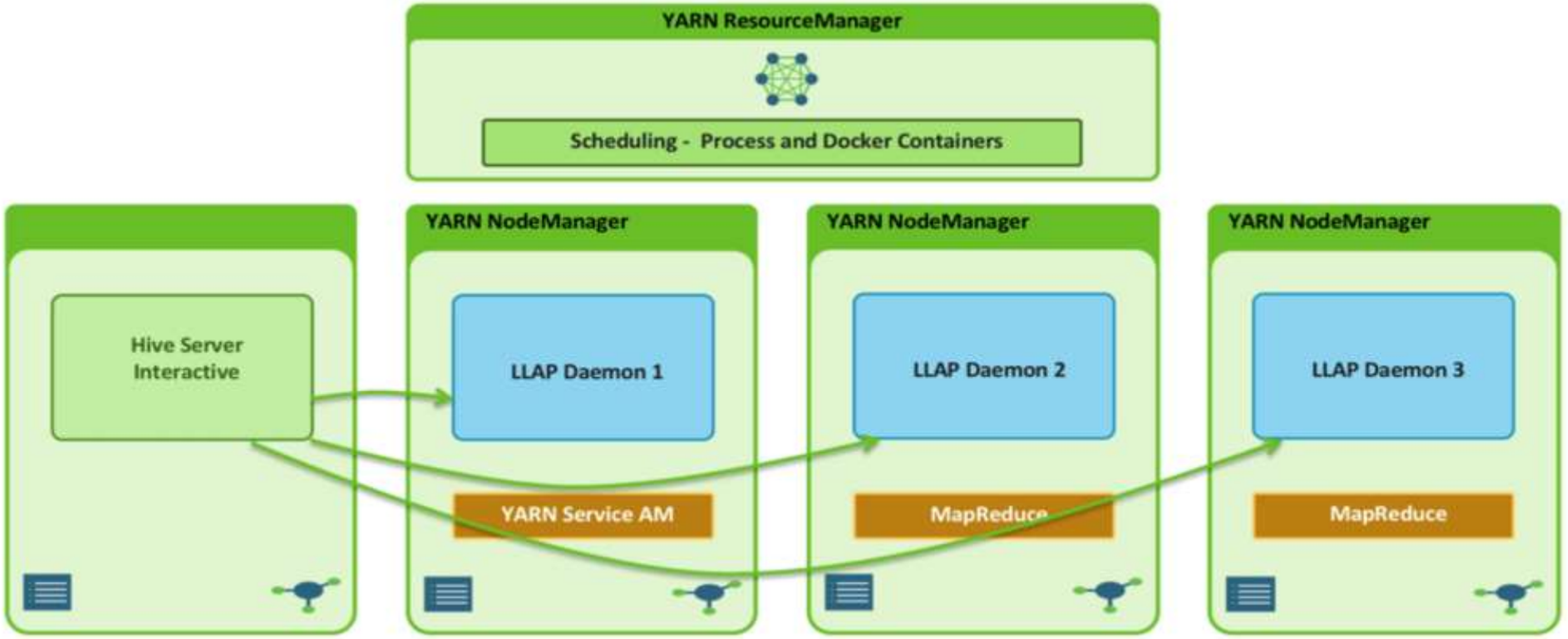

HORTONWORKS

How to run a new service in YARN ?



LLAP on YARN

- **Apache Hive LLAP** is a key long running application
  - Used for query processing
  - Designed to run on a shared multi-tenant YARN cluster



HORTONWORKS

Application Timeout – YARN-3813

- Controlling running time of workloads in YARN
- Define lifetime for an application anytime for YARN to manage.
- “Give me resources for this app/service but kill it after 15 days”

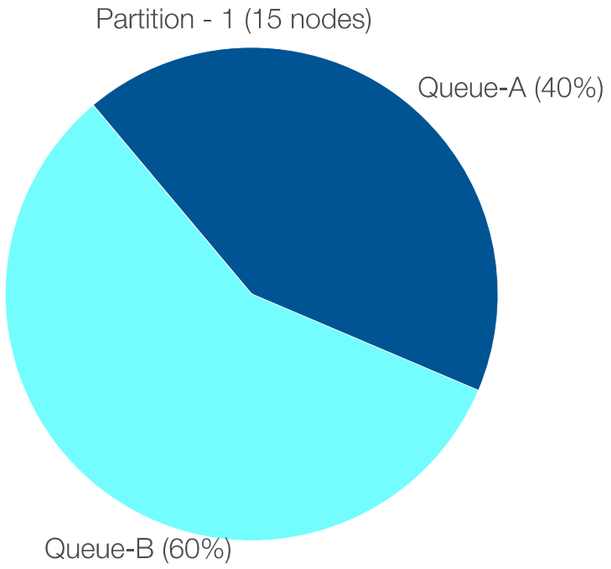
Reservation-based Scheduling: If You’re Late Don’t Blame Us! - Carlo, et al. 2015

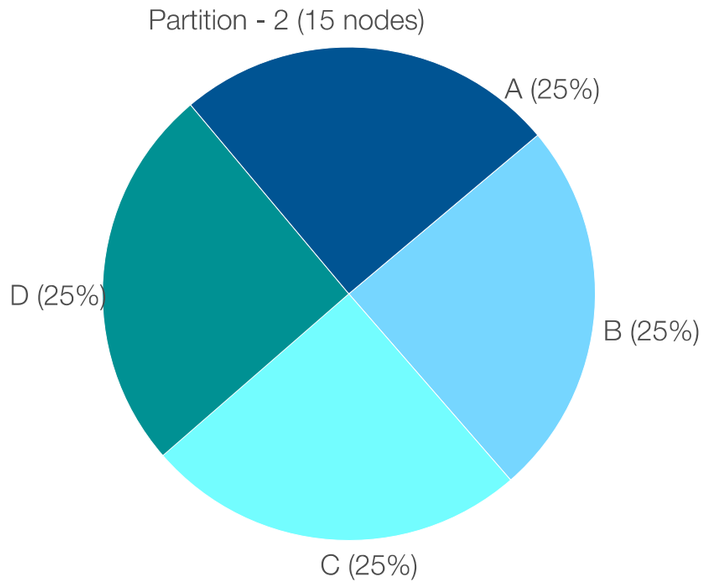
HORTONWORKS

Apache Hadoop 3.2 and beyond

Node Attributes (YARN-3409)

- “Take me to a node with JDK 10”
- Node Partition vs. Node Attribute
- Partition:
  - One partition for one node
  - ACL
  - Shares between queues
  - Preemption enforced.
- Attribute:
  - For container placement
  - No ACL/Shares on attributes
  - First-come-first-serve





Node 1
os.type=ubuntu
os.version=14.10
glibc.version=2.20
JDK.version=8u20

Node 2
os.type=RHEL
os.version=5.1
GPU.type=x86_64
JDK.version=7u20

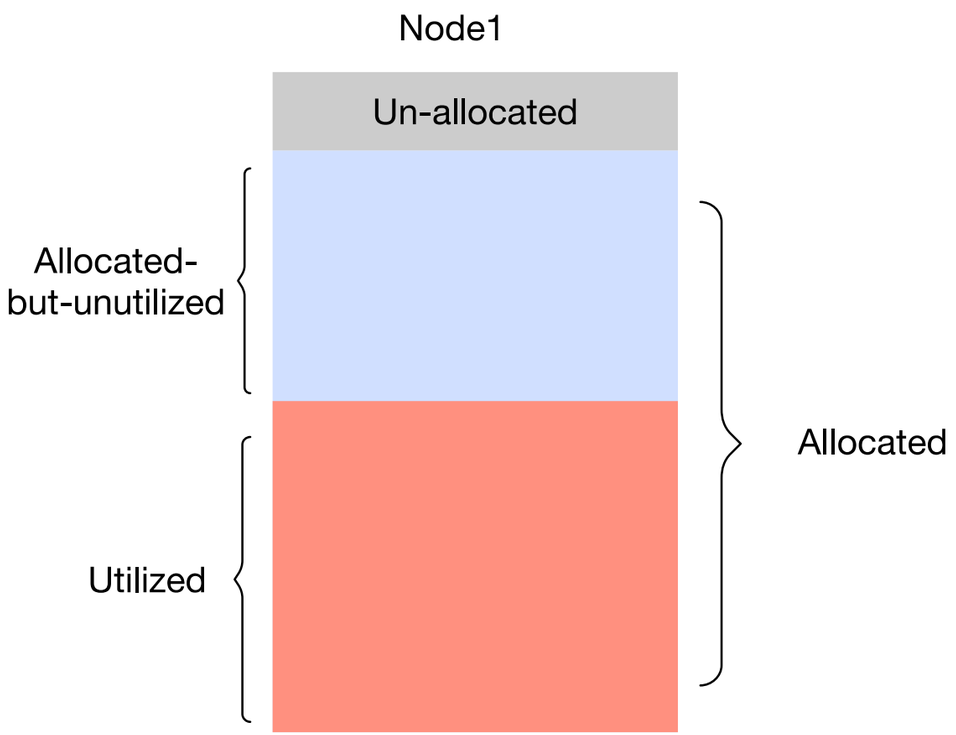
Node 16
os.type=windows
os.version=7
JDK.version=8u20

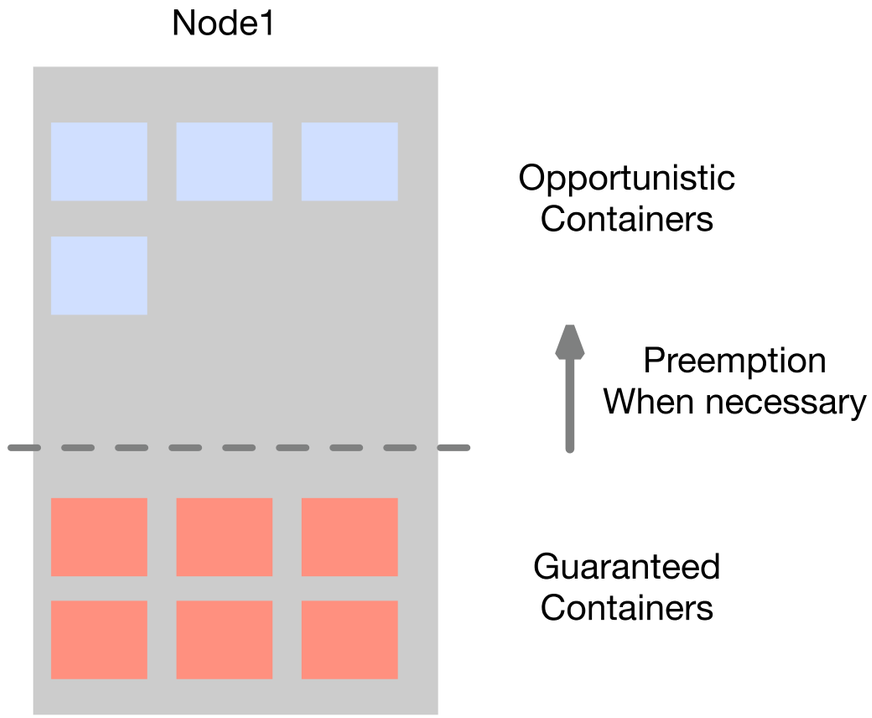
Node 17
os.type=SUSE
os.version=12
GPU.type=i686
JDK.version=7u20

HORTONWORKS

Container overcommit (YARN-1011)

- Every user says “Give me 16GB for my task”, even though it’s only needed at peak
- Each node has some allocated but unutilized capacity. Use such capacity to run opportunistic tasks
- Preempt such tasks when needed





Auto-spawning of system services (YARN-8048)

- “Start this service when YARN starts”
- “initd for YARN”
- System services is services required by YARN, need to be started during bootstrap.
  - For example YARN ATSv2 needs Hbase, so Hbase is system service of YARN.
  - Only Admin can configure
  - Started along with ResourceManager
  - Place spec files under *yarn.service.system-service.dir FS path*

```
SYSTEM_SERVICE_DIR_PATH
|---- sync
|       |--- user1
|       |      |---- service1.yarnfile
|       |      |---- service2.yarnfile
|       |--- user2
|       |      |---- service3.yarnfile
|       |      ....
|       |
|---- async
|       |--- user3
|       |      |---- service1.yarnfile
|       |      |---- service2.yarnfile
|       |--- user4
|       |      |---- service3.yarnfile
|       |      ....
|       |
```


HORTONWORKS

TensorFlow on YARN (YARN-8220)

- Run deep learning workloads on the same cluster as analytics, stream processing etc!
- Integrated with latest TensorFlow 1.8 and has **GPU** support
  - Use simple command to run TensorFlow app by using Native Service spec file (Yarnfile)

    yarn app -launch distributed-tf <path-to-saved-yarnfile>
  - A simple python command line utility also could be used to auto-create Yarnfile

    python submit_tf_job.py

    --**remote_conf_path** hdfs:///tf-job-conf

    --**input_spec** example_tf_job_spec.json

    --**docker_image** gpu.cuda_9.0.tf_1.8.0

    --**job_name** distributed-tf-gpu

    --**user** tf-user

    --**domain** tensorflow.site

    --**distributed** --**kerberos**

HORTONWORKS

TensorFlow on YARN (YARN-8220)

Sample Yarnfile for TensorFlow job

```
{
  "name": "distributed-tf",
  "version": "1.0.0",
  "components": [
    {
      "name": "worker",
      "dependencies": [],
      "resource": {
        "cpus": 1,
        "memory": "4096",
        "additional" : {
          "yarn.io/gpu" : {
            "value" : 1
          }
        }
      },
      "launch_command": "cd /test/models/tutorials/image/cifar10_estimator && python cifar10_main.py --data-dir=hdfs://default/tmp
      "number_of_containers": 1
    }
  ],
  "kerberos_principal" : {
    "principal_name" : "test-user@EXAMPLE.COM",
    "keytab" : "file:///etc/security/keytabs/test-user.headless.keytab"
  }
}
```


HORTONWORKS


Questions?

Thank you

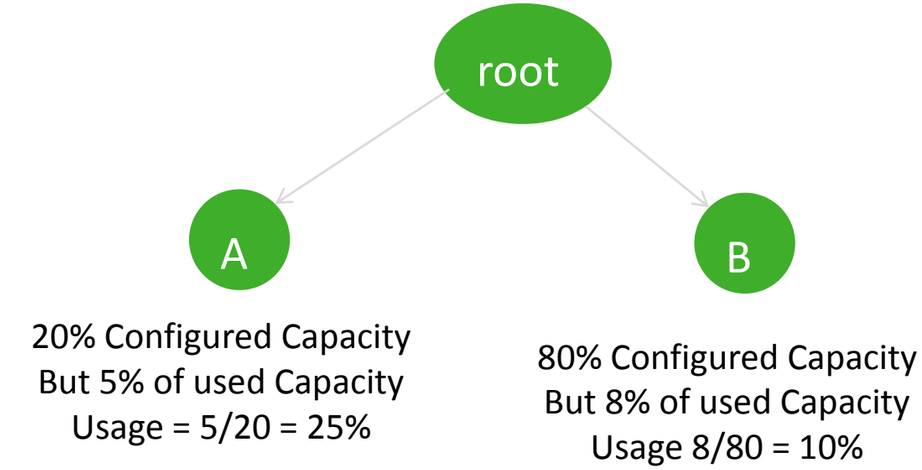

Apache Hadoop 2.8/2.9
HORTONWORKS

Application priorities / Queue priorities

- **YARN-1963**
- Allocate resource to important apps first within a Leaf Queue

Higher priority ➔ Lower priority

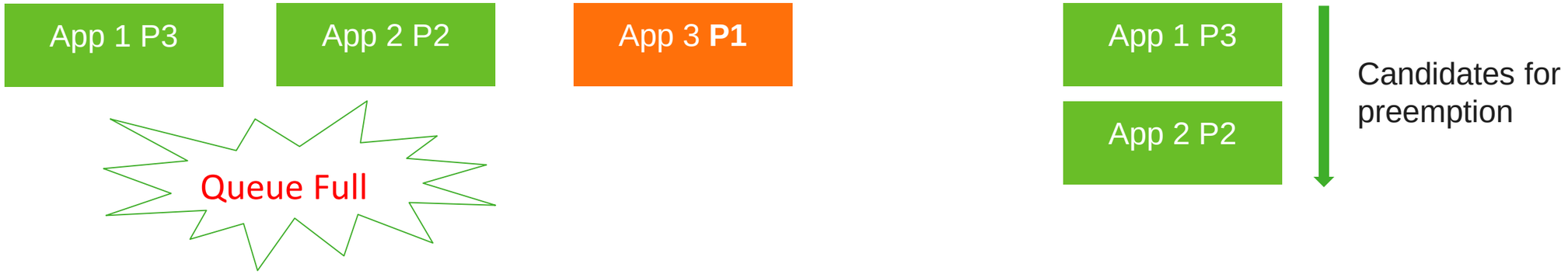
- **YARN-5864**
- Today give to the least satisfied queue first
- With priorities, Give to the highest priority queue first (for important workload).



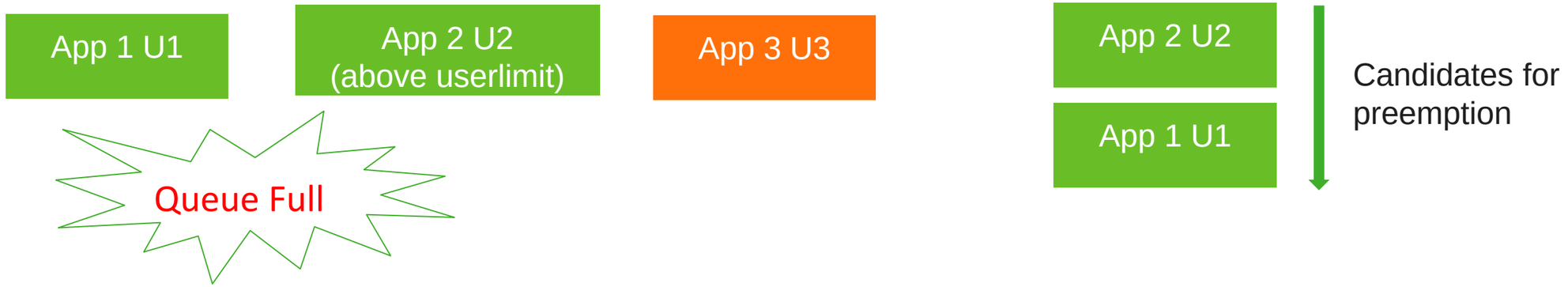
HORTONWORKS

Preemption within a queue – YARN-4945

- **Between apps of different priorities**



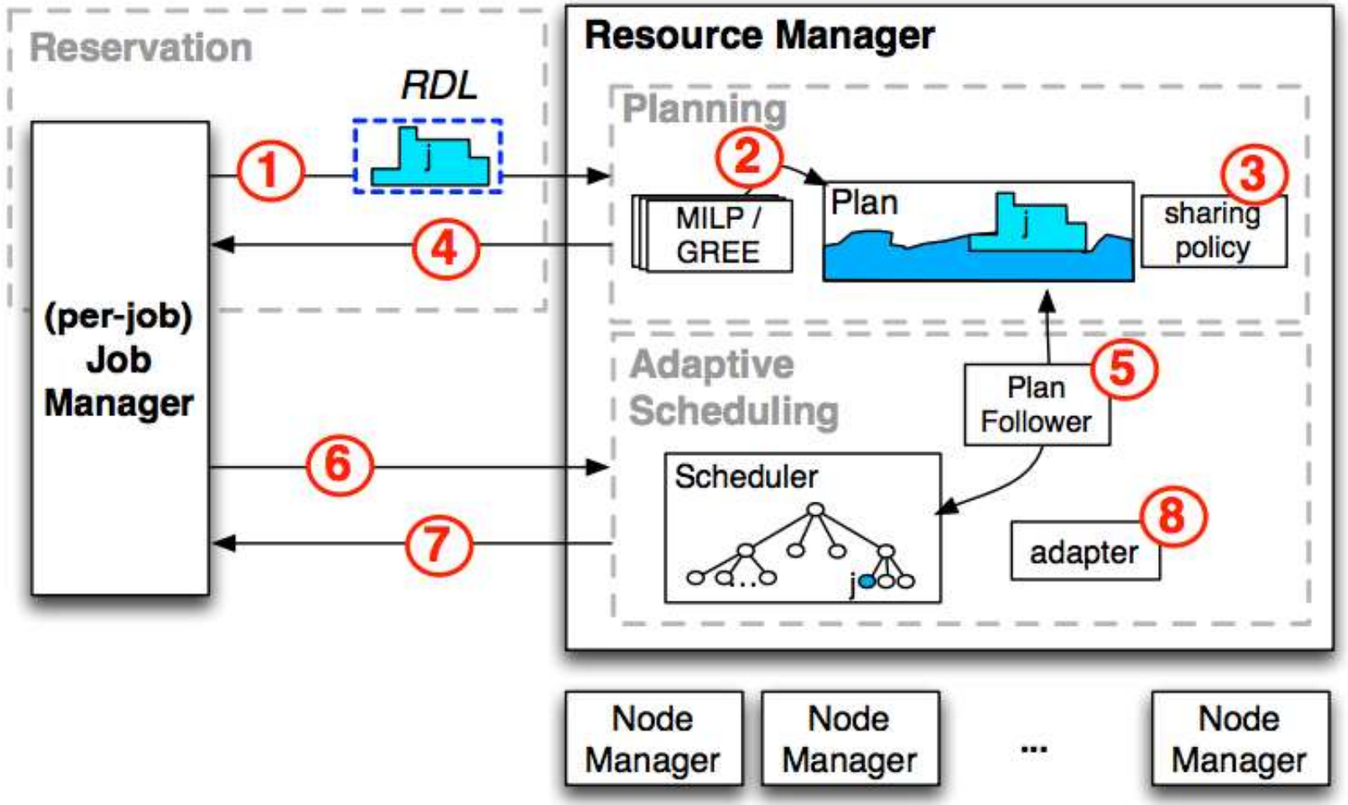
- **Between apps of different users**



HORTONWORKS

Reservations – YARN-1051

- *"Run my workload tomorrow at 6AM"*
- Persistence of the plans with RM failover: YARN-2573



Reservation-based Scheduling: If You're Late Don't Blame Us! - Carlo, et al. 2015