从YARN到Kubernetes
如何应对资源管理及作业调度的挑战

杨巍威
wwei@apache.org

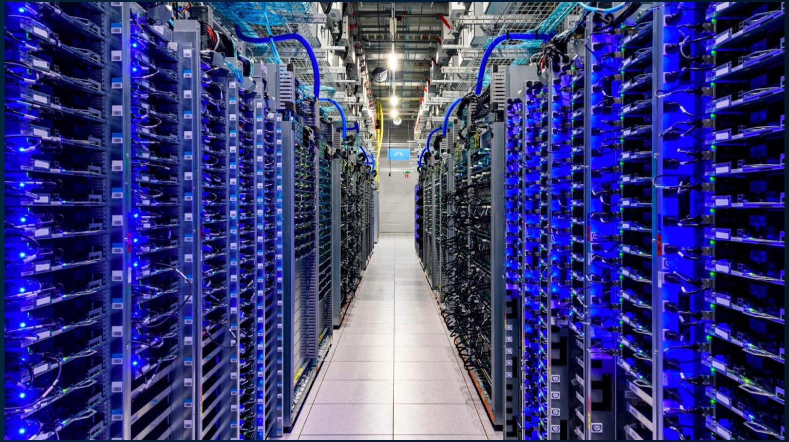
Weiwei Yang

Apache Hadoop Committer & PMC member
Apache YuniKorn (Incubating) PPMC member
Tech lead @ Cloudera
❤️ Open-Source

Apache Hadoop YARN

The resource management tool for “Hadoop”

- Natively supports Big Data workloads, MR, Spark, Tez, Flink, etc
- Majorly for batch workloads, lately provides service support
- Large scale, high throughput
- Mature for multi-tenancy
- Advanced scheduler: Fair/Capacity scheduler



Kubernetes

“Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation.”

KEY MERITS

- Separate “Compute” and “Storage”
- Containerization
- Unified platform for On-prem & Cloud
- Blooming ecosystem

Big Data on Kubernetes

Motivation

- An unified infrastructure for private, public and hybrid cloud
- Containerized workloads
- Flexible network, monitoring, storage, visualization
- Elasticity, cost saving
- Unified and simplified dev ops

Big Data on Kubernetes Today

Cloudera CDP

“CDP delivers powerful self-service analytics across hybrid and multi-cloud environments, along with sophisticated and granular security and governance policies that IT and data leaders demand.”

OpenData Hub

“Open Data Hub is a blueprint for building an AI as a service platform on Red Hat's Kubernetes-based OpenShift® Container Platform and Ceph Object Storage. It inherits from upstream efforts such as Kafka/Strimzi and Kubeflow, and is the foundation for Red Hat's internal data science and AI platform.“

Google Dataproc

“The launch of Cloud Dataproc on Kubernetes is significant in that it provides customers with a single control plane for deploying and managing Apache Spark jobs on Google Kubernetes Engine in both public cloud and on-premises environments.”

Challenges

Separate Compute and Storage

Decouple compute and storage, without sacrificing efficiency. Achieve the maximum agility to scale up and down computes.

Resource Management

The need of fine-grained resource management, pursue the balance between resource sharing and efficiency.

Job Scheduling

Lack of job level scheduling capability, service-originated resource scheduler doesn’t satisfy the complex big data scenarios.

Challenges Cont’d

Resource Management

- Namespace is flat (NO tree structure)
- NS Resource quota is calculated on admission phase
- When resource quota is exhausted, pods will be rejected (client-side-failure)
- No way to preempt resources from the other namespace
- No resource fairness between namespaces

Job scheduling

- No job ordering
- No job queuing
- No job priority
- No job level preemption

YuniKorn

Apache YuniKorn (Incubating) is a light-weight, universal resource scheduler for container orchestrator systems. It provides fine-grained resource sharing for various workloads efficiently on a large scale, multi-tenant, and cloud-native environment.

YuniKorn Features

Hierarchy Queues

Provides fine-grained control over resources for different tenants. Built-in with fairness support. By leveraging the min/max queue capacity, it can define how elastic it can be in terms of the resource consumption for each tenant.

Job Scheduling

YuniKorn queues jobs in resource queues and schedules them with respect to a certain ordering policy. It allows users to configure customized ordering policies for different use cases, it also provides job level fairness per queue.

Essential Capabilities

Built in with essential scheduling capabilities and ensures high performance at the same time. YuniKorn supports features like gang scheduling, resource reservation, preemption, these features empowers running big data & AI workloads on K8s.

Cloud-Native

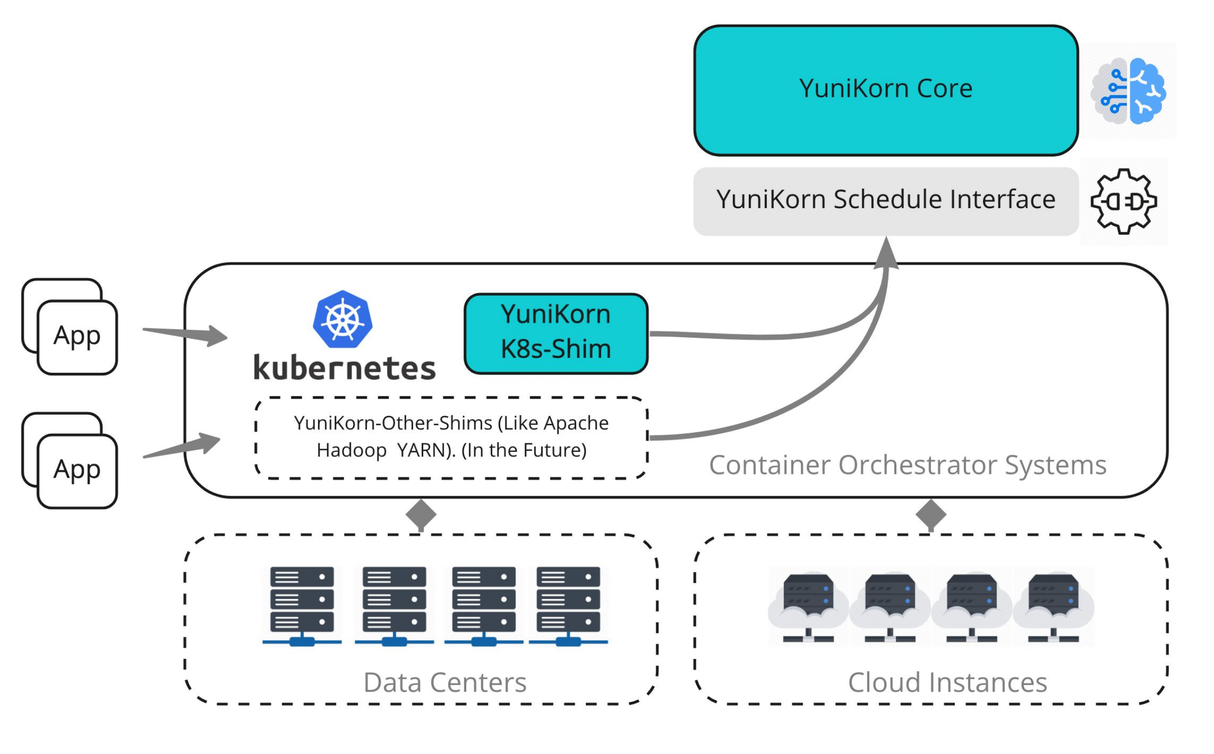
YuniKorn is designed to be lightweight and highly extensible. It works with cluster-autoscaler to scale up & down compute pools. It is a stateless service and it can be easily deployed with helm, it can automatically recover from faults for reliability.

Architecture

DECOUPLED An abstraction of scheduler-interface to decouple the scheduler-core with the underneath platforms

SCHEDULING Built-in with advanced scheduling capabilities to support both batch and long-running workloads.

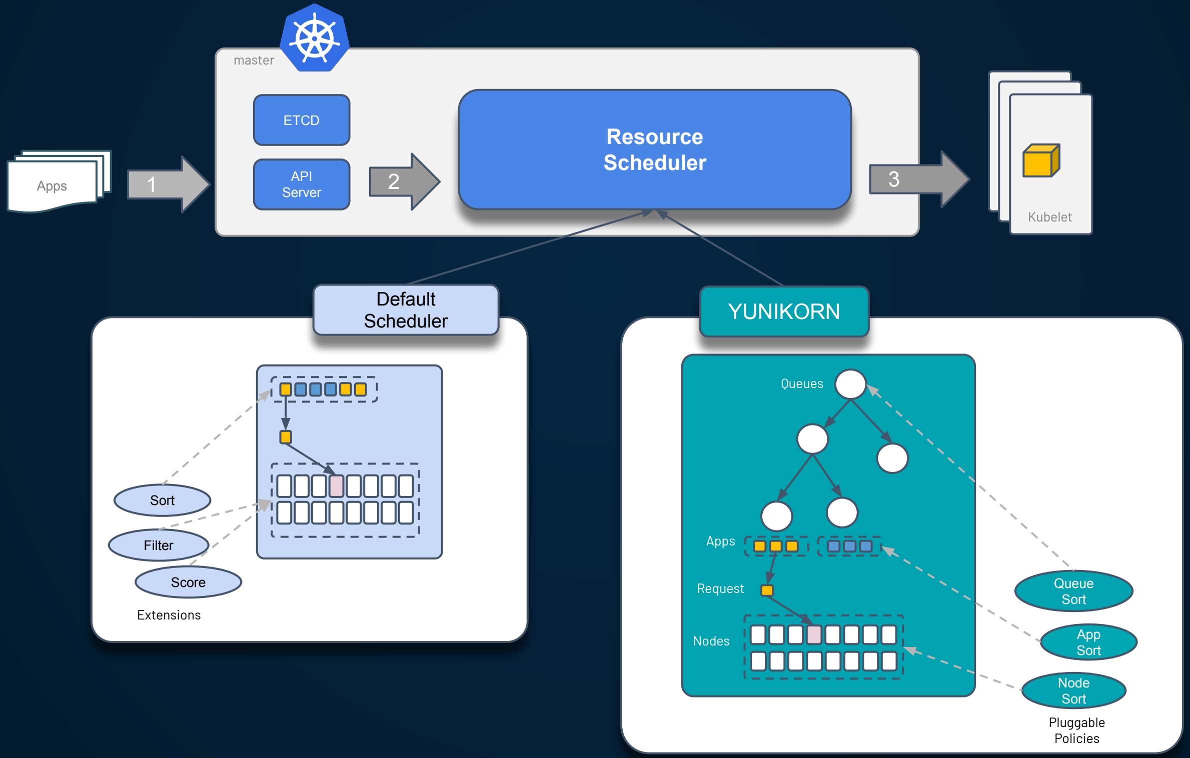
CLOUD-NATIVE Highly extendable, scalable, natively works on-prem and cloud.


YuniKorn Core
YuniKorn Schedule Interface
App
App
kubernetes
YuniKorn K8s-Shim
YuniKorn-Other-Shims (Like Apache Hadoop YARN). (In the Future)
Container Orchestrator Systems
Data Centers
Cloud Instances

Main Difference

Feature	Default Scheduler	YUNIKORN	Note
Scheduling at app dimension	⊘	✓	App is the 1st class citizen in YuniKorn, YuniKorn schedules apps with respect to, e,g their submission order, priority, resource usage, etc.
Job ordering	⊘	✓	YuniKorn supports FIFO/FAIR/Priority (WIP) job ordering policies
Fine-grained resource capacity management	⊘	✓	Manage cluster resources with hierarchy queues, queue provides the guaranteed resources (min) and the resource quota (max).
Resource fairness	⊘	✓	Inter-queue resource fairness
Natively support Big Data workloads	⊘	✓	The default scheduler is main for long-running services. YuniKorn is designed for Big Data app workloads, it natively supports Spark/Flink/Tensorflow, etc.
Scale & Performance	⊘	✓	YuniKorn is optimized for performance, it is suitable for high throughput and large scale environments.

Workflow


master
Apps
1
ETCD
API Server
2
Resource Scheduler
3
Kubelet
Default Scheduler
Sort
Filter
Score
Extensions
YUNIKORN
Queues
Apps
Request
Nodes
Queue Sort
App Sort
Node Sort
Pluggable Policies

Performance

Schedule 50,000 pods on 2,000/4,000 nodes.

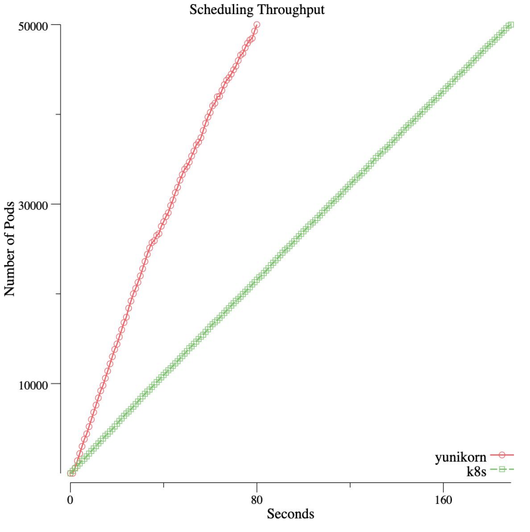
Compare Scheduling throughput (Pods per second allocated by scheduler)

Red line (YuniKorn)
Green line (Default Scheduler)

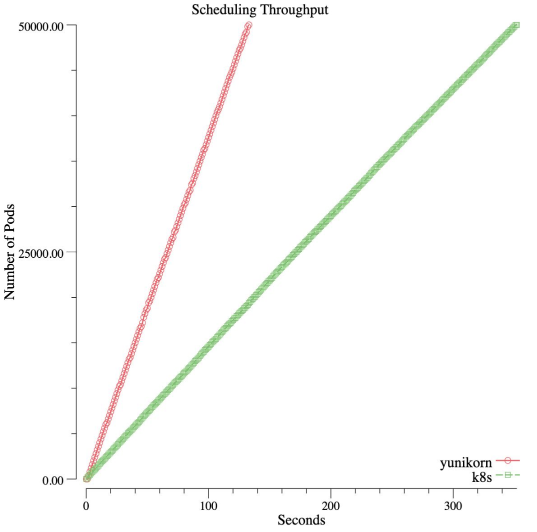
617 vs 263 ↑ 134%

373 vs 141 ↑ 164%

50k pods on 2k nodes

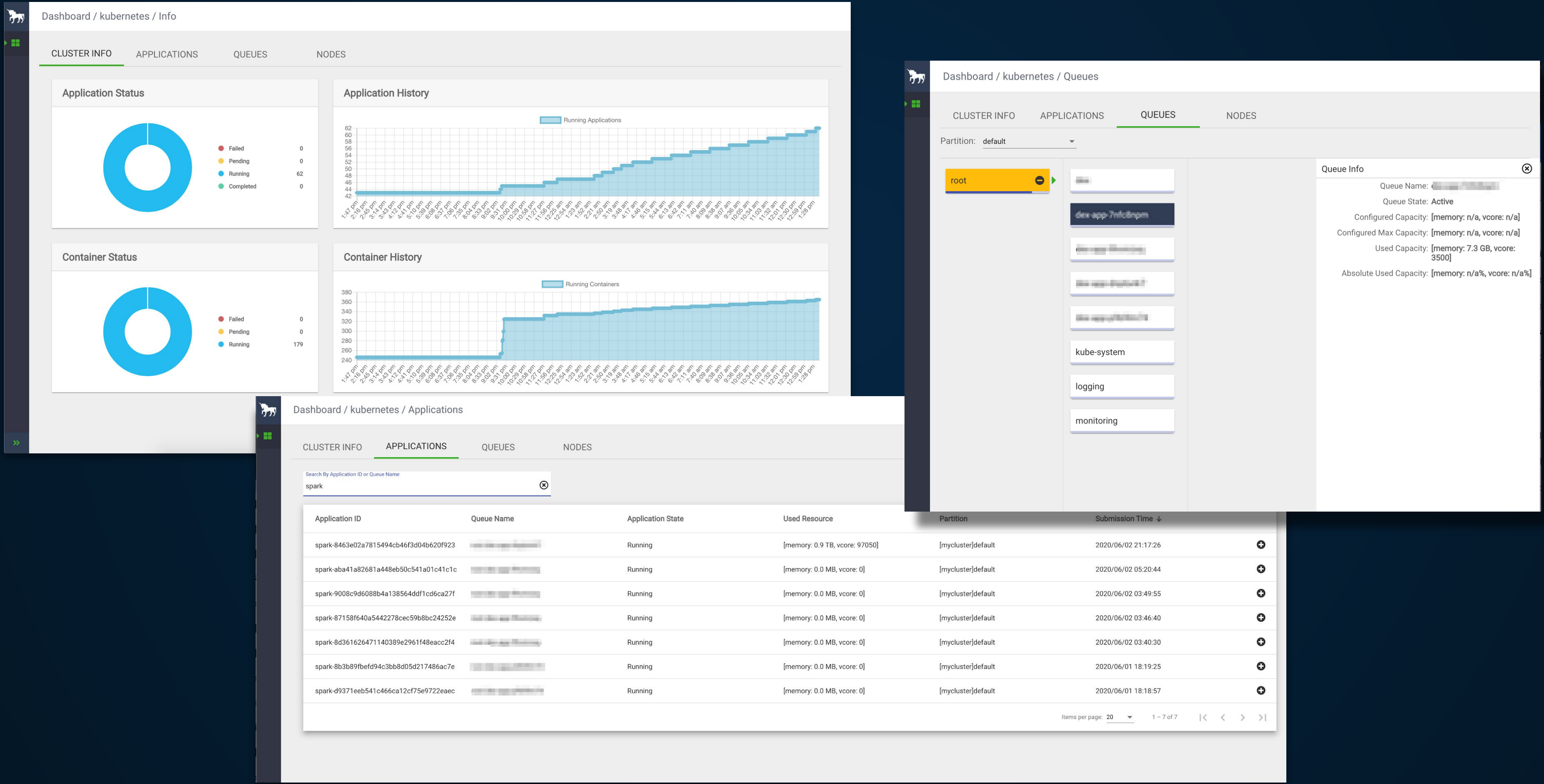


50k pods on 4k nodes



Detail report:
https://github.com/apache/incubator-yunikorn-core/blob/master/docs/evaluate-perf-function-with-Kubemark.md

Central Management UI



Dashboard / kubernetes / Info

CLUSTER INFO APPLICATIONS QUEUES NODES

Application Status

● Failed	0
● Pending	0
● Running	62
● Completed	0

Application History

Container Status

● Failed	0
● Pending	0
● Running	179

Container History

Dashboard / kubernetes / Queues

CLUSTER INFO APPLICATIONS QUEUES NODES

Partition: default

Queue Info

Queue Name: [redacted]
Queue State: Active
Configured Capacity: [memory: n/a, vcore: n/a]
Configured Max Capacity: [memory: n/a, vcore: n/a]
Used Capacity: [memory: 7.3 GB, vcore: 3500]
Absolute Used Capacity: [memory: n/a%, vcore: n/a%]

Dashboard / kubernetes / Applications

CLUSTER INFO APPLICATIONS QUEUES NODES

Search By Application ID or Queue Name
spark

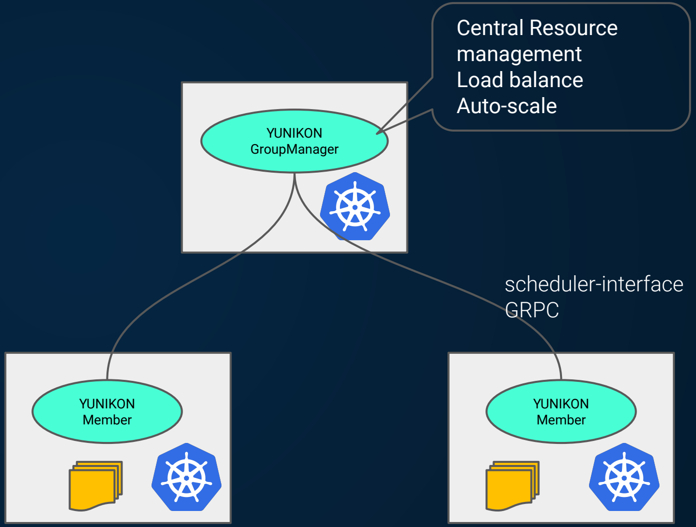
Application ID	Queue Name	Application State	Used Resource	Partition	Submission Time ↓	
spark-8463e02a7815494cb46f3d04b620f923	[redacted]	Running	[memory: 0.9 TB, vcore: 97050]	[mycluster]default	2020/06/02 21:17:26	⊕
spark-aba41a82681a448eb50c541a01c41c1c	[redacted]	Running	[memory: 0.0 MB, vcore: 0]	[mycluster]default	2020/06/02 05:20:44	⊕
spark-9008c9d6088b4a138564ddf1cd6ca27f	[redacted]	Running	[memory: 0.0 MB, vcore: 0]	[mycluster]default	2020/06/02 03:49:55	⊕
spark-87158f640a5442278cec59b8bc24252e	[redacted]	Running	[memory: 0.0 MB, vcore: 0]	[mycluster]default	2020/06/02 03:46:40	⊕
spark-8d361626471140389e2961f48eacc2f4	[redacted]	Running	[memory: 0.0 MB, vcore: 0]	[mycluster]default	2020/06/02 03:40:30	⊕
spark-8b3b89fbefd94c3bb8d05d217486ac7e	[redacted]	Running	[memory: 0.0 MB, vcore: 0]	[mycluster]default	2020/06/01 18:19:25	⊕
spark-d9371eeb541c466ca12cf75e9722eaec	[redacted]	Running	[memory: 0.0 MB, vcore: 0]	[mycluster]default	2020/06/01 18:18:57	⊕

Items per page: 20 1 – 7 of 7

Recent Work

Federation

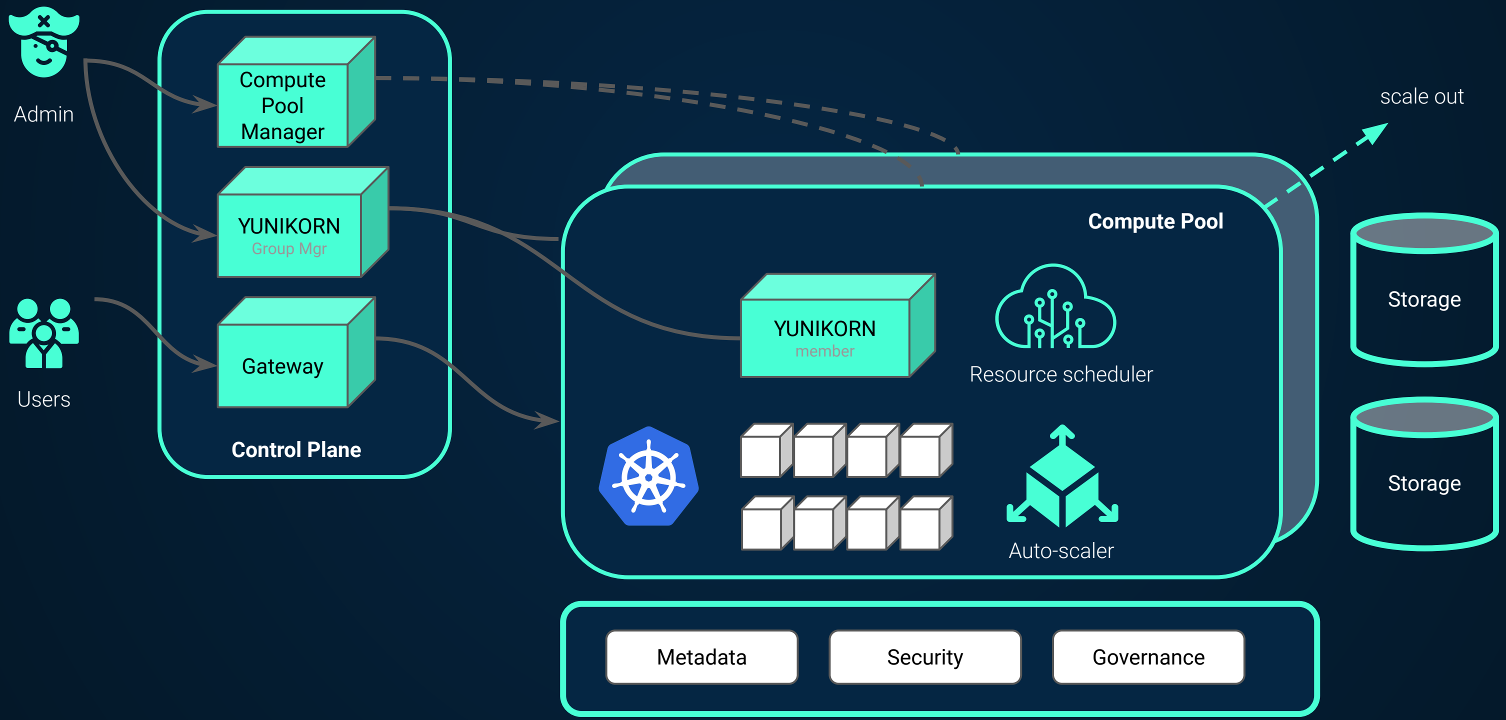
- Central gateway responsible for routing jobs with load balancing
- Central resource management via YUNIKORN
- Compute pools are dynamically scaled up/down



Central Resource management
Load balance
Auto-scale

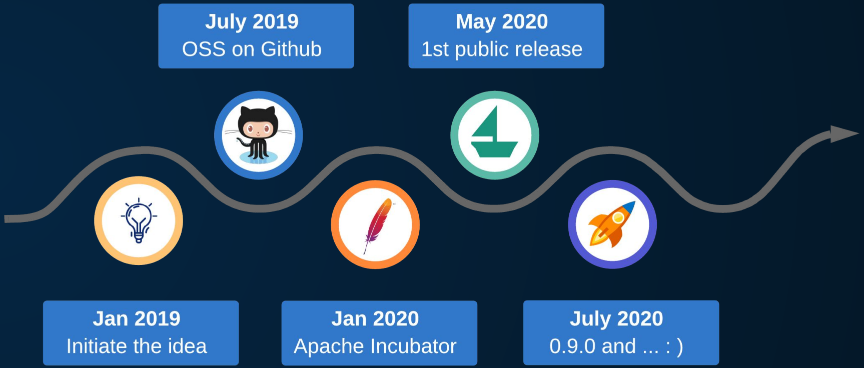
scheduler-interface
GRPC

Build the Big Data Infra



Project Status

Apache Incubator project: since Jan, 2020
Current Version: 0.9.0
Web-site: http://yunikorn.apache.org/
Community members: Alibaba, Apple, Cloudera, Databricks, LinkedIn, Microsoft, Nvidia



Jan 2019
Initiate the idea

July 2019
OSS on Github

Jan 2020
Apache Incubator

May 2020
1st public release

July 2020
0.9.0 and ... :)

YuniKorn 技术交流群

Valid until 9/30 and will update upon joining group

THANKS!

如果对 YUNIKORN 技术感兴趣请扫描入群
或者发邮件至 **dev@yunikorn.apache.org**