APACHE YUNIKORN ENHANCED SCHEDULING IN THE CLOUD

Wilfred Spiegelenburg
Manikandan Ramaraj
AGENDA

Why Apache YuniKorn

Architecture

Queues: hierarchies and quotas

Gang Scheduling

User and Group Quotas

Roadmap
WHY APACHE YUNIKORN

Advanced scheduling requirements

Workload Queueing

Gang Scheduling

Application Sorting
AGENDA

- Why Apache YuniKorn

**Architecture**

- Queues: hierarchies and quotas
- Gang Scheduling
- User and Group Quotas
- Roadmap
ARCHITECTURE

Deployment models

• **STANDARD:**
  - Custom scheduler
  - Replaces *default* scheduler

• **PLUGIN:**
  - Kubernetes Scheduling Framework (API)
  - Replace or augment limited functionality
PLUGIN ARCHITECTURE

Extension points

Pod Scheduling Context

Sort

PreFilter
Filter
PreScore
Score
Normalize Score
Reserve
Permit

Pick a Pod from scheduling queue

Reserve a Node for the Pod in Cache

PostFilter

problematic

Bind Pod to Node

WaitOnPermit
PreBind
Bind
PostBind

Binding Cycle

Scheduling Cycle
• **Multiple** initiatives around batch and HPC schedulers
  • Apache YuniKorn
  • Volcano
  • Kueue
  • Armada
• Scheduler plugins for **specific** point solutions
• Fragmentation concerns: KubeCON NA panel discussion

• Pre-enqueue plugin
  – KEP-3521: Pod Schedule Readiness

• Pre-emption behaviour: **priority** only
  – K8s has: do not preempt other pods during scheduling
  – Batch needs: do not preempt the tagged pod while running
AGENDA

Why Apache YuniKorn

Architecture

Queues: hierarchies and quotas

Gang Scheduling

User and Group Quotas

Roadmap
Hierarchical Model

- **root**
- **tenant**
- **non-yunikorn**
- **unlimited**
- **limited**

Namespaces defined in k8s
Hierarchical Model

- **root**: unlimited
- **tenant**: resource limits based on registered nodes
- **non-yunikorn**: K8s namespace quota, resource limit 75 GB / 75 CPU
- **unlimited**: pods
- **limited**: pods

Resource limits:
- **root**: unlimited
- **tenant**: resource limits based on registered nodes
- **non-yunikorn**: resource limit 75 GB / 75 CPU
- **unlimited**: pods
- **limited**: pods
Hierarchical Model

- **root**
  - **FAIR**
  - **tenant**
    - **FAIR**
    - **non-yunikorn**
      - **pods**
    - **unlimited**
      - **FIFO**
      - **pods**
    - **limited**
      - **pods**
AGENDA

- Why Apache YuniKorn
- Architecture
- Queues: hierarchies and quotas
  - Gang Scheduling
- User and Group Quotas
- Roadmap
GANG SCHEDULING

Gang example: Spark

Spark Application

Driver

gang spec: 1 driver, 3 executors

placeholders

YuniKorn
GANG SCHEDULING

Gang example: Spark

Spark Application

Driver
gang spec: 1 driver, 3 executors

driver

gang 1

gang 2

gang 3

placeholders

YuniKorn
GANG SCHEDULING

Gang example: Spark

Spark Application

Driver
gang spec: 1 driver, 3 executors

gang 1

gang 2

gang 3

placeholders

YuniKorn
GANG SCHEDULING

Gang example: Spark

Spark Application

Driver
gang spec: 1 driver, 3 executors

executor
executor
executor

placeholders

YuniKorn
AGENDA

Why Apache YuniKorn
Architecture
Queues: hierarchies and quotas
Gang Scheduling

User and Group Quotas

Roadmap
USER AND GROUP INFORMATION

Two options

Authenticated user:
- User information based on authenticated user
- Set by admission controller on pod creation

Annotation on pod:
- User information provided on pod creation
- Check & override by admission controller
USER AND GROUP INFORMATION

Two options

Authenticated user:
• User information based on authenticated user
• Set by admission controller on pod creation

Annotation on pod:
• User information provided on pod creation
• Check & override by admission controller
USER AND GROUP QUOTAS

Queues vs User & Groups

- Queue quota: root to leaf
  - Integrated in scheduling cycle
  - No quota available queue will be skipped

- User and group: leaf to root
  - Request fits in queue headroom
  - Last check before confirming node placement
AGENDA

Why Apache YuniKorn

Architecture

Queues: hierarchies and quotas

Gang Scheduling

User and Group Quotas

Roadmap
Roadmap
With all appropriate caveats...

Apache YuniKorn 1.2
- Application limits
  - Maximum running applications per Queue
- User and Group quotas
  - Implementation started for:
    - User retrieval
    - Usage tracking
  - Enforcement design in final stage

Future release
- Priority support
  - Early design stage
- Pre-emption
  - Obsolete code removed (YuniKorn 1.1)
  - Design starting
Q & A
THANK YOU