Using Mesos Schedulers with Amazon EC2 Container Service

Ryosuke Iwanaga

Solutions Architect, Amazon Web Services Japan

July 2016, LinuxCon+ContainerCon Japan
Agenda

• What's the scheduler for Amazon ECS
• Introduction of Apache Mesos and its schedulers
• How the ECS Scheduler Driver works
• Demo
• Case studies for custom schedulers
Amazon EC2 Container Service (ECS)

Container Management at Any Scale

Flexible Container Placement

Integration with the AWS Platform
Components of Amazon ECS

**Task**
Actual containers running on Instances

**Task Definition**
Definition of containers and environment for task

**Cluster**
Fleet of EC2 instances on which tasks run

**Manager**
Manage cluster resource and state of tasks

**Scheduler**
Place tasks considering cluster status

**Agent**
Coordinate EC2 instances and Manager
How Amazon ECS runs Task

Scheduler

Manager

Task Definition

Task

Agent

Cluster
Schedulers
What is a Scheduler?

• Determine desired state
• Check against current state
• Perform action
Optimistic shared state model
Amazon ECS: Scheduling

• Each scheduler periodically queries the current cluster state

Scheduler A  
Scheduler B  
Copy of cluster state

Cluster
Amazon ECS: Scheduling

- Each scheduler allocates tasks on the cluster
- Each scheduler updates the current cluster state

Run a task

Run a task
Amazon ECS: Scheduling

- If the resource is already claimed, the request will be rejected

Run a task on the same resource

$\Rightarrow$ Transactional
Amazon ECS: Scheduling

- Shared state optimistic scheduling
- All schedulers can see the current cluster state at all times
Key points for ECS Schedulers and Manager

Good isolation between Schedulers and Manager
   Single resource manager – Multiple schedulers
   Scalable & faster

ECS built-in Schedulers
   Run Task: Batch job
   Service: Long running application
Sometime you need Custom schedulers

Different resource, for specific task

Priority Queue

Want to have custom auto scaling

=> ECS allows you to add any custom schedulers in parallel
Apache Mesos

http://mesos.apache.org/
What's Apache Mesos?

Started at UC Berkeley

High utilization for diverse frameworks; Hadoop, MPI, etc.

Resource Offering
Mesos Architecture

http://mesos.apache.org/documentation/latest/architecture/
Resource Offering

http://mesos.apache.org/documentation/latest/architecture/
Mesos Schedulers

Marathon: for Long running application

Chronos: for Batch job with dependency

Apache Aurora: for Both long running and batch, as well as quota, DSL, etc.
How Marathon works

- **Task**
- **Slaves**
- **Master**
- **Marathon**
- **Mesos Scheduler Driver**
ECS Scheduler Driver

https://github.com/awslabs/ecs-mesos-scheduler-driver
ECS Scheduler Driver

Driver which can talk to ECS from any Mesos Scheduler

Written in Java

Without Mesos cluster

Control ECS cluster by Marathon, Chronos, etc.
ECS Scheduler Driver with Marathon

- ECS Scheduler Driver
- Manager
- AWS SDK for Java
- Marathon
- Amazon ECS
- Cluster
- Task
- Agent
Inside of ECS Scheduler Driver

To create resource offers:
  List / DescribeContainerInstances

To launch/kill tasks:
  Start / StopTask

To poll task status:
  List / DescribeTasks

AWS SDK for Java
How to use it

Clone the repository and build it to JAR file

Modify the scheduler to use ECSSchedulerDriver class instead of MesosSchedulerDriver

Or, override MesosSchedulerDriver class at runtime to ECSSchedulerDriver by bytecode enhancement
Use case

Same functionality between AWS and On-premise

Easy to get leverage from Third party schedulers
e.g. Marathon, Chronos, Aurora, etc.
Demo
Demo: ECS Scheduler Driver with Marathon

Cluster → Agent → Amazon ECS → Manager

Marathon → ECS Scheduler Driver → Marathon LB

Task
Write your scheduler for ECS

Example: Start Task on the least task running instance

def getInstanceArns(clusterName):
    containerInstancesArns = []
    # Get instances in the cluster
    response = ecs.list_container_instances(cluster=clusterName)
    containerInstancesArns.extend(response['containerInstanceArns'])

    # If there are more instances, keep retrieving them
    while response.get('nextToken', None) is not None:
        response = ecs.list_container_instances(
            cluster=clusterName,
            nextToken=response['nextToken']
        )
        containerInstancesArns.extend(response['containerInstanceArns'])

    return containerInstancesArns
Example: Start Task on the least task running instance

def startTask(clusterName, taskDefinition):
    startOn = []
    # Describe all instances in the ECS cluster
    containerInstancesArns = getInstanceArns(clusterName)
    response = ecs.describe_container_instances(
        cluster=clusterName,
        containerInstancesArns=containerInstancesArns
    )
    containerInstances = response['containerInstances']
Example: Start Task on the least task running instance

# Sort instances by number of running tasks
sortedContainerInstances = sorted(
    containerInstances,
    key=lambda containerInstances: containerInstances['runningTasksCount']
)

# Get the instance with the least number of tasks
startOn.append(sortedContainerInstances[0]['containerInstanceArn'])

# Start a new task
response = ecs.start_task(
    cluster=clusterName,
    taskDefinition=taskDefinition,
    containerInstances=startOn,
    startedBy='LeastTasksScheduler'
)
Case studies of custom schedulers
Amazon Personalization

Generating Recommendations at Amazon Scale with Apache Spark and Amazon DSSTNE

Recommendation for products at Amazon scale

CPU for data analytics, preparation
  => Apache Spark on Amazon EMR

GPU for train and predict neural network model
  => Amazon DSSTNE on Amazon ECS

Notebook for common entry point
  => Zeppelin on Amazon EMR
Architecture overview
Using ECS Scheduler Driver

To support task queue for GPU jobs

Current ECS does not have the native queue system

Leverage Mesos schedulers to get this easily
Coursera

http://www.slideshare.net/AmazonWebServices/cmp406-amazon-ecs-at-coursera-a-generalpurpose-microservice
https://www.youtube.com/watch?v=a45J6xAGUvA
Take the world's best courses, online.

What would you like to learn about?

or browse catalog

14,867,758 learners • 1,291 courses • 121 partners

Newest Specializations

- Strategic Business Analytics
  ESSEC Business School

- Hotel Management: Distribution, Revenue and... 
  ESSEC Business School

- Java Programming: An Introduction to Software 
  Duke University

- Data Warehousing for Business Intelligence 
  University of Colorado System

- Data Analysis and Interpretation 
  Wesleyan University

- Desenvolvimento e Design de Aplicativos para iPhone 
  Universidade Estadual de Campinas

- Business Strategy 
  University of Virginia

- Excel to MySQL: Analytic Techniques for Business 
  Duke University
What We Built: Iguazú

- Batch Job Scheduler for Amazon ECS
  - Immediately
  - Deferred (run once at X time)
  - Scheduled recurring (cron-like)
- Programmatically accessible internally via our standard APIs and clients
- Named for Iguazú falls
  - World’s largest waterfall by volume
  - We hope Iguazú handles a similar volume of jobs
Iguazú: Architecture

- Devs
  - Iguazú Admin
  - Iguazú Scheduler
  - Iguazú Frontend

- Users
  - Services
  - Cassandra

- ECS API
  - ECS Workers
  - SQS
### Iguazú: Developer / Ops User Interface

#### Iguazú Scheduled Jobs

<table>
<thead>
<tr>
<th>ID</th>
<th>Cron Expression (UTC)</th>
<th>Next Invocation</th>
<th>Family</th>
<th>Job Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>JQIGfMwFqW55cSXd</td>
<td>0 30 18 1/1 * ? *</td>
<td>Sep 23, 2015 11:30 AM PDT</td>
<td>sessionstart</td>
<td>sessionstart</td>
</tr>
<tr>
<td>wxzCVhPpxewwMe0</td>
<td>0 8 1/1 * ? *</td>
<td>Sep 23, 2015 1:00 AM PDT</td>
<td>lapsednewuserpush</td>
<td>lapsednewuserpush</td>
</tr>
<tr>
<td>cnuuzbA6algLsuGV</td>
<td>0 22 1/1 * ? *</td>
<td>Sep 23, 2015 3:00 PM PDT</td>
<td>notifyexpiredvc</td>
<td>notifyexpiredvc</td>
</tr>
<tr>
<td>FHB0YTvEBTlvFSP</td>
<td>0 20 1/1 * ? *</td>
<td>Sep 23, 2015 1:00 PM PDT</td>
<td>paymenttransactionsettlement</td>
<td>paymenttransactionsettlement</td>
</tr>
</tbody>
</table>
Programming Assignments at Coursera

- NVIDIA CUDA
- Scala
- RAILS
- Python
- JS
- MATLAB
- C Programming Language
- Spark
What We Built: GrID

• Service + architecture for grading programming assignments

• Builds on Amazon ECS and Iguazú

• Named for Tron’s “digital frontier”
  • Backronym: Grading Inside Docker
High-level GrIcD Architecture

Learners

GrIcD

Iguazu

Production Acct

S3 Bucket

VPC Firewalls

ECS API

GrIcD Grading Account

Grading Machines
One more thing...
Collect Container Logs via awslogs

Amazon ECS

Amazon CloudWatch Logs → Amazon S3

Amazon CloudWatch Logs → Amazon Kinesis

Amazon CloudWatch Logs → AWS Lambda

Amazon CloudWatch Logs → Amazon Elasticsearch Service
Collect Container Logs via awslogs

• Supported Docker Logging Drivers
  • json-file, syslog, journald, gelf, fluentd, splunk, awslogs
  • stdout/stderr outputs are automatically sent by the driver

• awslogs sends logs to Amazon CloudWatch Logs
  • Log group for a specific service
  • Log stream for each container

• Amazon CloudWatch Logs => Other services
  • Search, Filter, Export to Amazon S3, Send to Amazon Kinesis, AWS Lambda, Amazon Elasticsearch Service
Search logs with Kibana on Amazon ES
Conclusion

Amazon ECS is managed cluster management service

ECS exposes all cluster state so that any type of schedulers can easily control it in parallel

   via List / Describe * APIs

ECS Scheduler Driver is one of reference implementations
Thank you!