To Infinity and Beyond
Handling heterogenous container clusters in AWS
www.scout24.com
AWS Summit | Feb 2019 | Christine Trahe, Cloud Platform Engineering
Containerized Applications running on AWS
Enabling product engineers to focus on delivering value to Scout24 consumers and customers.

Platform Engineering, Scout24
Platform Setup
Multi-Account Strategy
Multi-Account Strategy
AWSTemplateFormatVersion: '2010-09-09'
Description: >
  Example infinity service

Resources:
  Service:
    Type: 'Custom::InfinityService'
    Properties:
      ServiceToken: 'arn:aws:lambda:eu-west-1:000000000000:function:angus'
      ImageName: 111111111111.dkr.ecr.eu-west-1.amazonaws.com/MyImage:latest
      ServiceName: MyNewService
      ContainerPort: 9000
      Memory: 256
      HealthCheckPath: /health

    Tags:
      - service: MyNewService
      - usecase: my-usecase
      - segment: my-segment
      - owner: my-team-email-distribution-list
      - team: my-team
      - stage: dev|prod
      - vertical: s24
What the user sees
Under the hood: Centralized compute infrastructure
Under the hood: Logging and Monitoring
What the user sees
Learnings

• Services affecting each other
• CloudFormation Application deployment
• Tooling around ECS
Services Affecting Eachother
But it doesn’t scale to Infinity just yet...

Load Balancer response time (latency) increases, causing real time customer impact

Too much CPU consumption on the container instances
Sharing resources not ideal...

Container Instance (EC2)

8 CPU Cores (Shared)

Reservation: 4
Usage: ~50% (2)
Container A

Usage (Bursting): 300% (6)
Reservation: 2
Container B

Reservation: 1
Usage: ~10% (>1)
Container C

Total Reservation: 7
Total Usage: 8
Application differences
<table>
<thead>
<tr>
<th>Category 1</th>
<th>Category 2</th>
<th>Category 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Configuration</td>
<td>Freely Configurable</td>
<td>Freely Configurable</td>
</tr>
<tr>
<td>Calculated based on Memory</td>
<td>Shared and is Burstable</td>
<td>Not Shared. Only Available for your container</td>
</tr>
<tr>
<td>Slow Scaling. &gt; 150% Average CPU</td>
<td>Moderate Scaling. &gt; 100% Average CPU</td>
<td>Fast Scaling. &gt; 50% Average CPU</td>
</tr>
</tbody>
</table>

Categorizing Services to their needs
Deploying Applications through CloudFormation
It’s CloudFormation stacks all the way down…
Stuck in Rollback State
Deploying directly with EKS K8S API
Tooling around ECS
Cross Account Permissions for a Service
Many open source tools for K8S
Key Takeaways
- What level to abstract
- Standards and best practices

- Higher level of abstraction
- Heterogenous workloads
- Large open source community
- AWS abstracts complexity

- Not for application deployment (API calls, AWS CodeDeploy)
- Infrastructure deployment
- Platform abstraction

- Lower level building block
- Non-heterogenous workloads
- Smaller community
Thank you! Questions?

Christine Trahe

AWS Summit Feb 2019 | Cloud Platform Engineering, Scout24

https://jobs.scout24.com/#/department-positions/PlatformEngineering

We are hiring!
jobs.scout24.com