DevOpsBot

Jaykumar Patel, Raviteja Kommalapati, Shraddha Kabade, Srinivasa Prasad Sunnapu
Software Engineering Department, San Jose State University
California, USA

***Abstract* - DevOpsBot aims to ease the life of an AWS Engineer by automating repeated operations encountered in application development or performance testing life cycle using a conversational solution. This document outlines the solution approach, architecture, components and integrations involved.**

Index terms - Alexa custom skill, AWS Lambda, AWS EC2, AWS Java SDK, Amazon CloudWatch, Spring Boot, PaaS DevOps.

I. Introduction

DevOpsBot automates routine AWS operations performed by AWS developer or performance engineer like deploying application or load scripts, scaling up / down clusters, querying performance metrics etc. by taking conversational approach with Alexa. AWS engineer can get this done by having a conversation with Alexa, instead of connected to computer and performing these operations on AWS console. This solutions involves integration of different component like Alexa Custom Skill, AWS Lambda Intents interpreter and EC2 provision request handler using AWS SDK.

II. Architecture

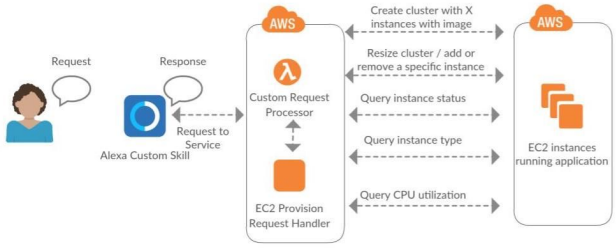

Fig.1 System Infrastructure and Architecture

A. Alexa Custom Skill

Alexa skill kit provides a way to generate custom interaction model by adding new abilities. To build a custom skill, we provided invocation name, various intents for our actions, many sample utterances. Also for our cluster names we created custom slot types and also used existing slot (AMAZON.NUMBER). To handle user requests, aws lambda endpoint is provided.

B. Lambda Intent Interceptor

When user makes a request, alexa service identifies the intent and routes the request to the appropriate service. The lambda code inspects the request, get the slot values and makes a call to EC2 provision handle. Cloudwatch was used to monitor AWS lambda.

C. EC2 Provision Request Handler

It is the REST API built on spring boot which handles requests from Intent Interceptor and responds with status response text to be prompted by Alexa. This API uses AWS SDK to handle EC2 provision requests, query likes instance type, cluster type, CPU utilization etc. For performing monitoring on the instances, instances are enabled with detailed monitoring using cloudwatch APIs which offers provision to query various performance parameters like CPU, Network etc.

D. AMI baked with Application / Test Scripts

It is a pre-baked AMI with application or load test scripts with appropriate tags given to each AMI. User will refer to these tags while conversing with Alexa to perform specific intent using that AMI or instance tagged with image tag.

III. System Interaction

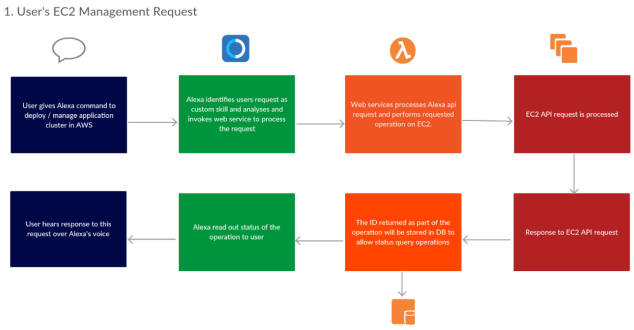

Fig.2 User's EC2 Management Request

1

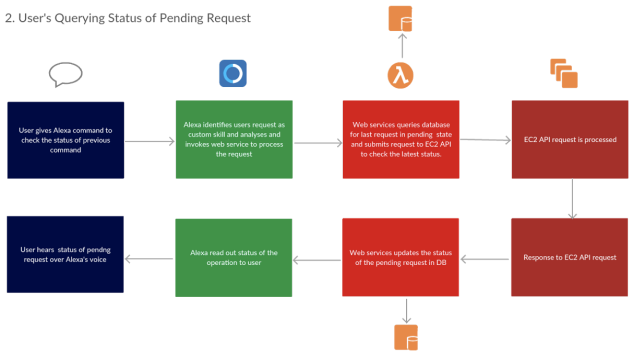

Fig.3 User's Querying Status of Pending Requests

Use Invocation name to start: "Open Ops Helper".
Sample utterances for each action are listed below.

1. Scale up cluster:
   Utterance: Scale up <clusterName> cluster by <numbeOfInstances> VM instances
   Provision-API-Endpoint:
   http://<host>/provision/hosts/create?image_tag=<clusterName>&count=<numberOfInstances>
2. Scale down cluster:
   Utterance: Scale down <clusterName> cluster by <numbeOfInstances> VM instances
   Provision-API-Endpoint:
   http://<host>/provision/hosts/state?host_tag=<clusterName>&action=terminate&count=<numberOfInstances>
3. Resize cluster:
   Utterance: Resize <clusterName> cluster to <numberOfInstances> instances
   Provision-API-Endpoint:
   http://<host>/provision/hosts/state?host_tag=<clusterName>&action=resize&count=<numberOfInstances>
4. Find pending requests:
   Utterance: What is the status of my previous request
   Provision-API-Endpoint:
   http://localhost:8080/provision/hosts/query?param=status
5. Query size of cluster:
   Utterance: What is the size of <clusterName> cluster
   Provision-API-Endpoint:
   http://localhost:8080/provision/hosts/query?host_tag=<clusterName>&param=size
6. Query VM instance type:
   Utterance: What is the instance type of <clusterName> cluster
   Provision-API-Endpoint:
   http://localhost:8080/provision/hosts/query?host_tag=<clusterName>&param=type
7. Query CPU utilization:
   Utterance: What is the average cpu utilization of <clusterName> cluster
   Provision-API-Endpoint:
   http://localhost:8080/provision/hosts/query?host_tag=<clusterName>&param=cpu

IV. Technologies

TABLE 1
List of technologies used

Name	**Version**
Spring Boot	2.0.0
NodeJS	8.10
Java	1.8
AWS SDK for Java	1.11.21
Alexa Skill Kit	1.0.0

V. Conclusion

Thus DevOpsBot enables AWS engineer to be able to multitask by automating time consuming tasks with simple interactions with Alexa.

The future work could involve expanding the scope of AWS operations and making DevOpsBot everybody's friend.

Acknowledgment

We are thankful and express sincerest regards to our guide, Prof. Rakesh Ranjan, for his valuable inputs, guidance, encouragement and cooperation throughout.

We would like to appreciate a special thanks to the Department of Software Engineering, San Jose State University for the support.

References

[1] AWS lambda Documentation [online]. Available : https://aws.amazon.com/lambda/
[2] AWS EC2 Documentation [online]. Available : https://aws.amazon.com/ec2/
[3] Amazon Alexa Skills kit [online]. Available: https://developer.amazon.com/alexa-skills-kit
[4] AWS SDK for Java Developer [online]. Available: https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/welcome.html
[5] Alexa Fact Skill Example [online]. Available: https://github.com/alexa/skill-sample-nodejs-fact